

important than ever before to get a head start on competition for jobs.

3. DETAILED EXAMPLE

Let us follow through for a moment how an actual job lead would be processed. The procedure is pretty much the same regardless of whether the lead is classified as teaching or administration on an elementary, secondary, two-year or four-year college level or falls into a general category such as accountant, computer programmer, etc.

3.1 The Job Vacancy. Let us assume that a request has come in from a high school principal who is looking for someone who:

- a. seeks a TEACHING position in SECONDARY education in FRENCH
- b. has a MASTER'S degree
- c. is willing to work in SYRACUSE, New York
- d. will be available for employment in DECEMBER
- e. is willing to accept \$10,000 per year
- f. has minimum related work experience of TWO years

A member of the Placement staff can very quickly type into the terminal keyboard the following message:

SEL POS T S FRE DEG MAS LOC SYR AVA DEC SAL 10 EXP 2

This is the shorthand notation describing the job. The operator could also enter a longhand version of the same search request as follows:

SELECT POSITION = TEACHING, SECONDARY, FRENCH

DEGREE = MASTERS, LOCATION = SYRACUSE, AVAILABILITY =

DECEMBER, SALARY = 10,000, EXPERIENCE = 2 YEARS

Obviously, the shorthand method is quicker and is the method used by the Placement Service staff. In either case, the computer will reply within seconds with a printed count or total of the number of persons on file who are interested in receiving that particular job lead--let's say the computer has identified 14 individuals.

Now there are a number of additional commands which can be executed against the group of 14 selected records. The LIST command will simply list the names of the 14 individuals. The TPRINT command will provide a brief printout of each individual indicating the registrant's name, temporary address, zip code, temporary telephone number, degree date, major field, availability date and maiden name if applicable. The PPRINT command is similar except that it will display the permanent address and telephone number.

For the most complete printout, the operator can execute the DUMP command which will print for each of the previously 14 selected records everything on file in a clearly formatted manner. Please note that once the computer has determined those records which meet the criteria of the SELECT or search command, none of the subsequent commands require a repeat of the search algorithm. The system retains the computer addresses of the 14 selected records which minimizes execution time of other commands.

The usual objective of a job search is to have the system automatically print the names and addresses of the selected group. The PLABELS command will print name and permanent address data

in a proper format for the 14 individuals. If the previous search had resulted in 84 candidates being selected, then 84 labels would print. You can very easily appreciate the very real savings realized in personnel time since manual searching is eliminated--so is the typing of envelopes. A sample printout of names and addresses is shown in Figure 2.

3.2 Other System Uses. All processing of job leads do not result in a mailing to candidates. Sometimes a lead is received by telephone, and the caller would like to have the names of likely candidates so that he can personally contact them immediately. Under these circumstances, the operator can excuse himself for several minutes, go to the terminal and execute the appropriate computerized search along with appropriate printout commands, return to the telephone and provide the caller with as much data as he would like on the selected candidates. This is not unusual.

Another common occurrence is for a company representative to call in person at the Placement Service office concerning job vacancies. Often he will leave with a computer printout of likely candidates prepared for him while he watched the staff member operate the terminal. The only data item he would not receive on any DUMP printout would be the minimum salary that the candidate would be willing to accept. For obvious reasons, the Placement Service does not want to make this information available to prospective employers; however, it is still used during the search process to select the job candidates.

4. TERMINAL COMMANDS

Any terminal language is made up of words which were agreed upon with the user during the design stage of the project as to what they mean and how they will be used.

4.1 The Verbs. These words are used as commands to cause the computer to execute selected program modules to arrive at desired results:

<u>Long Version</u>	<u>Short Version</u>	<u>Function</u>
SELECT	SEL	Build a file of selected individuals who meet the job criteria. The nouns are also used with this verb to fully describe the job.
LIST	LIS	Print only the names of the selected individuals
PPRINT	PPR	Print a brief report on each of the selected individuals using the permanent address
TPRINT	TPR	Print a brief report on each of the selected individuals using the temporary address
PLABELS	PLA	Print mailing labels for each of the selected individuals using the permanent address
TLABELS	TLA	Print mailing labels for each of the selected individuals using the temporary address
DUMP	DUM	Print a complete record dump except acceptable minimum salary for each of the selected individuals
COUNT	COU	Print the total number of active records on the Applicant Master File

4.2 The Nouns. These words are used to define the vacancy criteria and are used with the SELECT verb to form a complete command statement for searching the files.

<u>Long Version</u>	<u>Short Version</u>	<u>Function</u>
POSITION	POS	Used to indicate the position or job code in a search. For example: POS = REG (a registrar vacancy)
EXPERIENCE	EXP	Used to indicate the minimum experience acceptable to the employer. For example: EXP = 5 (five years required)
SALARY	SAL	Used to indicate the salary being offered for the vacancy. For example: SAL = 12 (candidates must be willing to accept \$12,000)
LOCALITY	LOC	Used to indicate the geographic location of the vacancy. For example: LOC = NEA (candidates must be willing to work in the New England states area)
AVAILABILITY	AVA	Used to indicate when the employer wants to fill the position. For example: AVA = IMM (candidates must be available to begin work immediately)
DEGREE	DEG	Used to indicate the minimum educational requirement for the job. For example: DEG = MAS (candidates must possess at least a master's degree)
NAME	NAM	Used to search the files by a name, either partial or complete. For example: NAM = JONES THOMAS (all candidates with that name will be selected)

4.3 Search Combinations. By using the SELECT verb and all of the nouns (except NAME) to describe the vacancy, the most complete and refined search can be made for qualified candidates. Only those persons who match on every criteria will be selected, all others excluded, even those who might qualify on five criteria but miss out on only one criteria will find themselves rejected from the selected group. However, it is not necessary to use all of the nouns with the SELECT verb. The following examples will illustrate the flexibility of the system:

- a. SELECT POSITION = TEACHING All teaching candidates will be selected regardless of the subject area, level of teaching or any other criteria.
- b. SELECT POSITION = TEACHING 4 All teaching candidates who want employment at a four-year institution will be selected.
- c. SELECT POSITION = TEACHING 4 ENGLISH All teaching candidates who desire employment at a four-year institution in the subject field of English will be selected.

You can see that as more criteria are added, the search becomes more refined and the selected candidate list more filtered; thus, it is possible to really zero in on the most qualified candidates and only inform them of the job lead. Those who are not fully qualified by the selection criteria will not be bothered by unnecessary mail and, of course, the prospective employer likewise will not be unnecessarily bothered. The system works well for all parties concerned. Just a few more examples will illustrate that it is not even necessary to use the POSITION noun on every search. It all depends on what type of information

you are seeking from the file.

- d. SELECT SALARY = 15 All candidates who are willing to accept a salary of \$15,000 will be selected regardless of any other criteria.
- e. SELECT AVAILABILITY = DECEMBER All candidates who are willing to begin work in December will be selected regardless of any other criteria.
- f. SELECT SALARY = 15 AVAILABILITY = DECEMBER All candidates who are willing to accept \$15,000 and can begin work in December will be selected. Both criteria must be met in order to be selected.
- g. SELECT NAME = PET All candidates whose names begin with the letters PET will be selected--this would include Peters, Peterson, Petrie, etc. If the entire name is known, it can be used to select just that individual and others who might have the same exact name.

4.4 Candidate Status. When a person accepts a position, he is requested to notify the Placement Service so that his computer record can be updated to an inactive status. The position he has accepted is not entered into the files. Only an inactive flag is set to exclude the person from any future searches thus reducing processing and mailing costs. Unfortunately, there is no way to enforce the request, and it is felt that a number of successful job applicants never do notify the Placement Service of their good fortune. The problem is somewhat minimized by terminating the use of the old file in September and starting a new file each October. A person must re-register to get back into the system--no records are carried forward.

5. IN CONCLUSION

As I indicated in my opening remarks, our Placement Service office has had a very good experience with the computerized

placement system. Job leads can be processed swiftly and accurately thus providing a very valuable service to our students and alumni. The system is easy to modify so that each summer prior to the start of a new processing year, we will implement reasonable changes to the computer system to further enhance the effectiveness of the service for the coming year.

Yes, we do firmly believe based on our experience that the SUNYA PLACEMENT SYSTEM is an INNOVATIVE SYSTEM which is a SOLUTION and not an ILLUSION.

STATE UNIVERSITY OF NEW YORK AT ALBANY APPLICATION FOR PLACEMENT

101

SEE BROCHURE FOR INSTRUCTIONS

USE PENCIL ONLY

001 NEW CHANGE
TYPE ENTRY

A01 SOCIAL SECURITY NO

A02 NAME (LAST FIRST M)

A03 LANGUAGE SKILLS

A06 APT OR DORM (TEMP)

A07 TEMP STREET ADDRESS

A08 AREA CODE TEMP PHONE NO

A09 CITY & STATE (TEMP)

A10 ZIP CODE

A11 INT OR RE-REG

A12 STATUS

A13 APT OR DORM (PERM)

A14 PERMANENT STREET ADDRESS

A15 AREA CODE PERM PHONE NO

A16 CITY & STATE (PERM)

A17 ZIP CODE

A18 COLLEGE OR UNIV IF CURRENT STUDENT

A19 MONTH EXPECTED DEGREE DATE

A20 COURSE MAJOR

A21 COURSE MINOR

A22 PRIN SUPT GUID TCH LIB OTH
CERTIFICATION

B01 MINIMUM ACCEPTABLE YEARLY SALARY

B02 BACH MAS ABD DOC
DEGREE REGISTERED FOR/HIGHEST HELD

B03 IMM DEC MAY AUG
AVAILABILITY

B04 LOP NYC HUD ALB ADM MOH BN SYR ROC BUF MYS
NEW YORK STATE GEOGRAPHIC AREAS

B05 MEA MAT SEA SOU NCE MWWE MUS USA
UNITED STATES GEOGRAPHIC AREAS

B06 OSE
OVERSEAS GEOGRAPHIC AREAS

APPLICANTS FOR EDUCATIONAL INSTITUTIONS

TEACH	B07	LEVEL	POSITION	EXP	B08	LEVEL	POSITION	EXP	B09	LEVEL	POSITION	EXP	B10	LEVEL	POSITION	EXP
ADM	B11	LEVEL	POSITION	EXP	B12	LEVEL	POSITION	EXP	B13	LEVEL	POSITION	EXP	B14	LEVEL	POSITION	EXP

GENERAL APPLICANTS

GEN	B15	POSITION	EXP	B16	POSITION	EXP	B17	POSITION	EXP	B18	POSITION	EXP

FORMAL HIGHER EDUCATION (MOST RECENT FIRST)

C01	INSTITUTION	C02	YEAR	C03	MAJOR	C04	MINOR	C05	DEG COMP
C06		C07		C08		C09		C10	
C11		C12		C13		C14		C15	
C16		C17		C18		C19		C20	

EMPLOYMENT HISTORY (MOST RECENT FIRST)

C21	INSTITUTION	C22	INCLUSIVE DATES BY YEAR	C23	POSITION TITLE
C24		C25		C26	
C27		C28		C29	
C30		C31		C32	

D01 APPLICANT SIGNATURE

D02 APPLICATION DATE

FIGURE 1

106

RETURN COMPLETED
FORM TO

THE
PLACEMENT SERVICE
PM 135 ADMIN BLDG
SUNYA
1400 WASHINGTON AVE
ALBANY N Y 12222

RISHEL, RITA L
30 HITCHINS RD
BALSTON SPA, NY 12023

RAOLIN, BEVERLY R.
APT 4B
29-08, 139, STREET
FLUSHING NY 11354

QUELL, LOUISE A
12 WINTER LA
LEVITON, NY 11756

HANAY, MARGARET L
BOX 11
WESTERLY, NY 12193

BAINGARTNER, HOWARD S
2 FISHER DR
MT VERNON, NY 10552

DILLON, JAMES EDWARD
64 BRISQI AVE
STATEN ISLAND, NY 10312

OASIS: AN ADMINISTRATIVE
DATA MANAGEMENT SYSTEM

Cheryl M. Traver
Associate Director of Operations
School of Business Administration
Georgia State University

OASIS - AN ADMINISTRATIVE DATA MANAGEMENT SYSTEM
Cheryl M. Traver, Associate Director of Operations
Georgia State University, School of Business Administration
University Plaza, Atlanta, Georgia 30303

In the last five years two major advances have been made toward the improvement of commercial data processing productivity: the File Management System and the Management Information System. The former is directed at reducing the effort in processing volume data, the latter at providing more flexible and timely information out of stored data. Both of these substantial needs are prevalent in most commercial environments. Unfortunately, the two types of system are normally mutually exclusive as designed and supported. Thus, the data processor must select one or the other or constantly transpose his data between two conflicting data structures. It should be possible to provide a system that combines the advantages of the File Management and Management Information Systems. OASIS, an Online Administrative Information System developed at Stanford University, is such a system.

1. THE CURRENT AVAILABILITY

In the last five years the commercial data processing industry has made several attempts to move out of the strictly batch-oriented mode of operation which the media of the card and the magnetic tape had produced. Broadly speaking, the two most notable changes were produced by the introduction of first the File Management System and then the Management Information System. The two systems have basically different goals and, consequently, differ in their characteristics.

1.1 The File Management System. The File Management System (FMS)* was designed to be basically batch-oriented. This does not mean that it will not operate via terminals; however, the design was definitely oriented toward

*MARK IV of Informatics, Inc., is a fine example of the FMS. WORK TEN of Honeywell Information Systems is another example.

processing a large proportion of the data in an entire data set or file. Traditional batch retrieval methods are usually employed; the file structures are sequential or employ one major index or key via which the data can be retrieved. The rules on subsetting or sorting the data are in general very strict, the only flexibility being afforded via standard system sorts or embedded sort verbs. The transaction/master file concept characterizes these systems. They are extremely record-oriented and basically designed to access each record in a data set once (or very few times) to retrieve and/or update all of the data within a record in a single pass. Consequently, access times are primarily dependent on the speed of accessing the device and are largely independent of the relative efficiency of the code which manipulates the I/O-bound routines. Heavy emphasis is usually placed on the size of the file. The power and flexibility are directed at reducing mundane programmer effort. Typically such systems provide for automatic performance of standard updating functions; transaction/master file logic is included for matching records. Capabilities for generating list reports and doing standard subtotalling are often advertised. The systems are definitely oriented toward the programmer, often employing shortcut languages and fill-in-the-blank methods to permit rapid production of programs.

1.2 The Management Information System. The Management Information System (MIS)*, on the other hand, is terminal (or retrieval) oriented, designed primarily to handle random requests for complex selections of data. Its emphasis on rapid retrieval required the development of many forms on non-standard file structures designed specifically for doing random retrievals in

*TDMS (TS/CDMS) of System Development Corporation and IMS of International Business Machines Corporation are examples of the MIS.

the minimum amount of time. Instantaneous response for very flexible selection criteria is of primary importance. Normally the final output encompasses very few records and very little data. Such systems often tend toward the integrated data base concept: the data are arranged within a structure so that ingenious interrelationships can be produced. Such systems, being concerned so specifically with retrieval speeds, do not place heavy emphasis on the physical size of the files. In fact, extra space is normally required to repeat data in numerous structures or to support pointers, chains, and indices to make the rapid location of data easy. These systems are "user oriented" in that they provide tools for the user himself to specify and produce queries and simple reports. Sometimes generalized facilities are also included which permit the simple maintenance of data. (Most such systems do not include convenient programmer interfaces for writing nonstandard programs.)

1.3 The Dichotomy. Both the FMS and the MIS provide worthwhile and necessary capabilities. The normal commercial data processing environment has the need for programming volume maintenance and retrieval functions and for permitting the flexible compilation of information from those data. Unfortunately, those very characteristics which make the FMS adept at handling large volumes of data restrict the retrieval flexibility whereas the complex file structures which allow the MIS that flexibility force maintenance functions to be exceedingly slow.

The alternatives for the data processing environment are not very promising. Conceivably, a given installation could purchase both types of system and use each for its designated purposes. Unfortunately, this would require transposing the data from one system to the other at frequent intervals. Even if this were possible, it would be expensive. The second alternative

would be to purchase just an FMS, which would make the programming burden less cumbersome but would not facilitate the handling of unanticipated requests.

The third alternative - to acquire only an MIS - is the most accepted today.

Such a system affords management with the type of information it wants but is normally extremely expensive in terms of batch computing time for volume maintenance and retrieval.

1.4 The Need. It is the conviction of the author that none of these alternatives should be necessary. It should indeed be possible to provide one single system which would combine the advantages of both types of system with none of the large disadvantages that each today forces. Such a system would require that the same data base be used effectively for both mass retrieval and maintenance functions and for random inquiry and terminal updates. The batch access and modify speeds must be reasonable; the storage overhead for online retrieval must not be exorbitant. The system must provide inquiry services and report definition/generation features which are directed at the level of the person who will be the user of these services - i.e., English-like language in a fairly flexible format.

1.5 A Solution. A system that combines the advantages of the File Management System and the Management Information System, yet attempts to avoid the inherent weaknesses of each, is not an unrealistic dream. In fact, one such system already exists and is operational at Stanford University. Called OASIS (Online Administrative Information System), the system was developed by Project INFO, a software team at Stanford University, under a grant from the Ford Foundation.

2. THE OASIS APPROACH

In 1968 the Project INFO team was formed to address the problems of a university administrative computing community. Its mission was to develop for Stanford University the means by which the information needs of the administrative environment could be satisfied. (The requirements of scientific and research computing were not addressed.) After a nine-month study of the systems currently available (or nearly available) it was decided that none could be combined or structured in such a manner that the university administrative needs could economically be satisfied. At that time the basic ideas for the design of OASIS were formulated. As the name, Online Administrative Information System, implied, OASIS was designed to support both "online" and batch activity concerned with normal "administrative information" requirements via an organized and efficient "system". Thus, OASIS is an attempt to span the gap between the File Management System and Management Information System.

2.1 The Components. OASIS is highly modularized to provide maximum flexibility and is composed of four logical components: an Executive Control program, the Terminal Services, the File Services, and the Generalized Services.

The Executive Control program performs traditional monitoring functions to handle the supervision and coordination of the operation of multiple user tasks. The monitor handles task scheduling and the allocation of memory to the tasks and to I/O buffers. It also contains a polling routine which performs communications between the CRT programs and the external user. A Command Language Processor handles simple requests between the user and the system. The OASIS Executive differs from other monitors mainly in the space allocation techniques used by the Linkage Services, since OASIS divides memory

into 2K increments in which the Generalized Services and user tailored services operate.

The Terminal Services provide communications between the user and the system. They are used by the Command Language Processor and the Generalized Services, and are available to tailored applications through standard high-level language CALL statements (as, the COBOL CALL).

The File Services handle all interactions between the Generalized and tailored services and the OASIS data base. The structure of that data base reflects the recognition of needs for both unanticipated, unstructured user requests and efficient volume retrieval and maintenance. Services are provided for accessing the data in various ways for the varying demands.

The Generalized Services were designed to handle those needs in the administrative environment which could most readily be put into a generalized format and could be utilized by the user community, thereby avoiding laborious programming effort. The Generalized Services consist chiefly of QUERY and the Terminal Report Generator. QUERY was designed to handle the unanticipated user requests. The service is available through the terminal and consists of free-form English-like sentences input by the user. Automatic editing and formatting provide pleasing displays with a minimum amount of user effort. The Terminal Report Generator, on the other hand, was designed to avoid some of the programming needed for the fairly standard reports the user so often requires. Report definitions are entered through the terminal, with a step-by-step user guided procedure, and several options are provided for report generation.

2.2 The File Services. The power of OASIS is derived primarily from its data management routines and its file structure. Based on the assumption that

both volume requirements and complex selection criteria for low-volume output are required, OASIS supports both direct record retrieval and index-like lookup capabilities. An OASIS data base is physically resident on one or more disk storage devices and consists of one or more files which may or may not be related according to their definitions in the OASIS Dictionary. An extensive set of backup and recovery facilities is an integral part of the system.

2.2.1 Record Structure. An OASIS file is a physical collection of variable length "records". All of the data for a given record are located in contiguous space, since it is assumed that the information in that record is somehow logically related and as such will often be accessed together. Each record may be physically subdivided into a sequence of "segments", each of which may in turn contain one or more "elements". Logically related elements may be defined together as a "group". File boundaries may be spanned by means of a pseudo-element called an "indirect reference". Figure 1 is a pictorial description of the OASIS file structure.

The various segments within a record facilitate highly variable length records and at the same time provide a means for efficiently accessing and updating logical subdivisions of the record. (Within one file there can be a maximum of 254 different segment types, each specified as optional or mandatory and single or recurring.)

The element, on the other hand, was designed for user flexibility and convenience. Although an integral part of a record, the element is not dependent on its physical location within the record. Associated with each element in the OASIS Dictionary are a name and an OASIS edit picture which determine the data characteristics and how they are to be edited for display

purposes. (The rules on edit pictures permit inclusion of almost any printable character.) These names and standard formats were designed as a convenience so that data might be obtained in a manner more meaningful to the layman than a numeric or structural description.

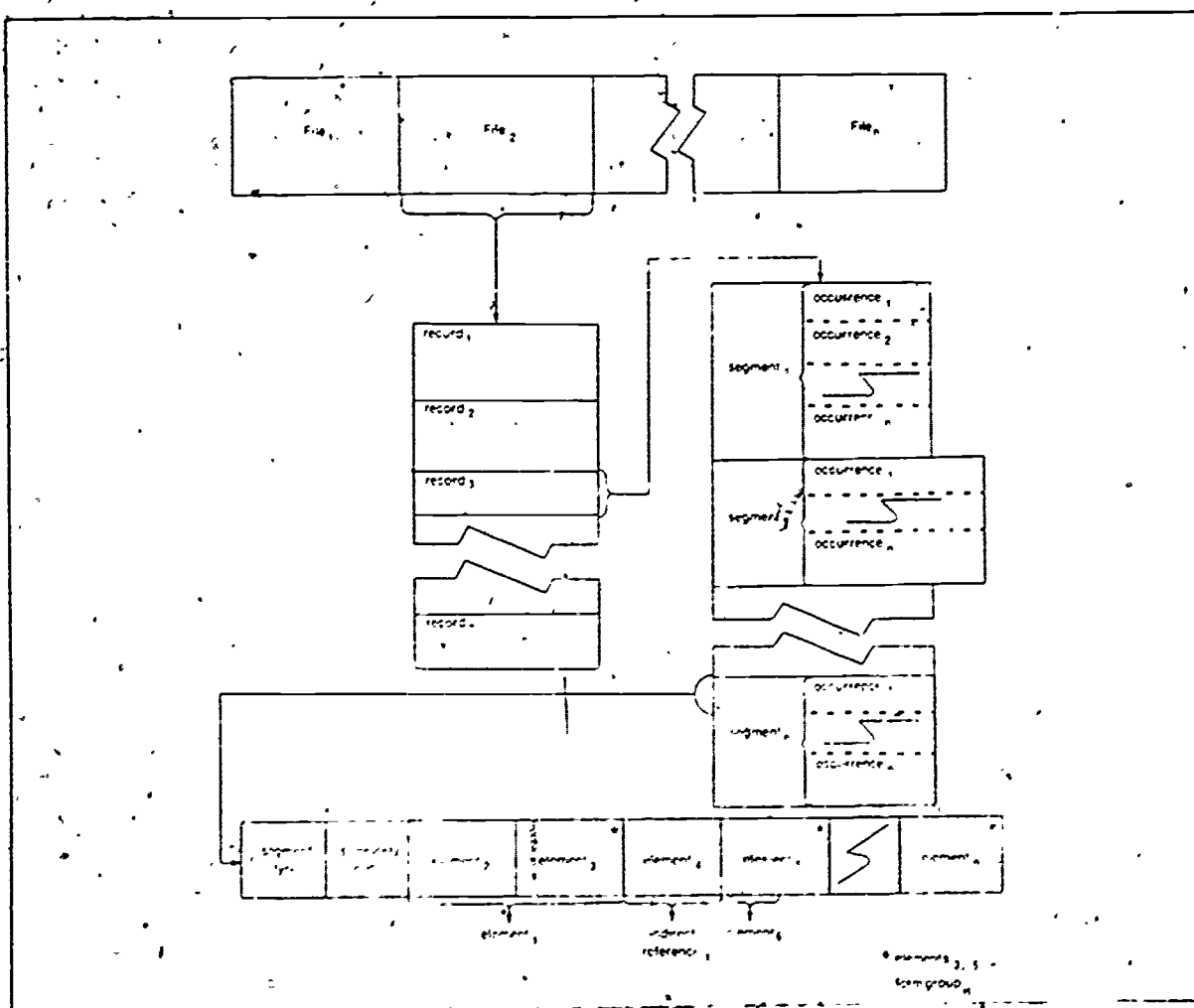


Figure 1. Oasis File Structure

2.2.2 The Services. The File Services routines reflect the OASIS record structure. There are services that deal with data according to the physical record layout:

GETSEG - Retrieves a particular occurrence (or all occurrences) of one segment type in a record, or retrieves an entire record in a file.

RPLSEG - Replaces a particular segment occurrence into a data record.

ADDSEG - Inserts a new physical segment occurrence into a data record.

DELSEG - Removes a particular segment occurrence from a data record, or an entire record from a file.

Other services are more user-oriented and less concerned with the record format:

GETDATA - Retrieves one occurrence of one or more elements, groups, and/or indirect references in a record, or all occurrences of one element.

RPLDATA - Changes the value of an element in a record.

GETVAL - Retrieves one, many, or all values which occur within the instances of an indexed element in a file.

These services, used in conjunction with OASIS routines for description and editing, may be used for data independence in a program. In fact, the Generalized Services do employ them in this manner.

2.2.3 Security. Because of the need for confidentiality where data are so readily available, OASIS provides a complex security structure. Each file, segment, and record within the OASIS data base has associated with it an access and a modify "code". All users of the system, whether they be terminal users or batch programs, are assigned passwords; each of which has an access and a modify "clearance". All of the OASIS File Services automatically check the clearances against the codes of the data which are being requested and

perform the service only if the security is valid. Security levels of three kinds are provided: a main hierarchy consists of 32 security levels (1-32). The 26 exclusive hierarchies, denoted A through Z and consisting of levels 1 through 8, are equal in security level and normally cover all or part of the main hierarchy but do not permit clearance into any other exclusive hierarchy. Absolute clearance is used for entry into all of the data in a given file. The hierarchies may be combined in any manner desired by the user to produce a specific security configuration.

2.2.4 Data Base Structure. In addition to those aspects of the OASIS file structure involving the subdivisions of records, OASIS includes facilities for the rapid retrieval and maintenance of those records. Associated with each file in an OASIS data base is a record number index (RNI). As each new record is added to a file, it is assigned a symbolic record number. The number itself has no significance; however, it represents the displacement into the RNI table where the actual physical location of the record is stored. When a record is relocated during maintenance activity, the change in physical location is recorded in only one place, the RNI. The significance of these symbolic record pointers becomes apparent when one considers the third and final component of an OASIS file.

A value index table (VIT) may be built for each element whose values are frequently used in retrieving data. Each VIT contains an ordered list of values existing for that element in the file. Associated with each value is a list of the RNIs of those records containing the particular value. OASIS automatically updates these VITs during all maintenance functions to the file.

Figure 2 illustrates the OASIS data base components.

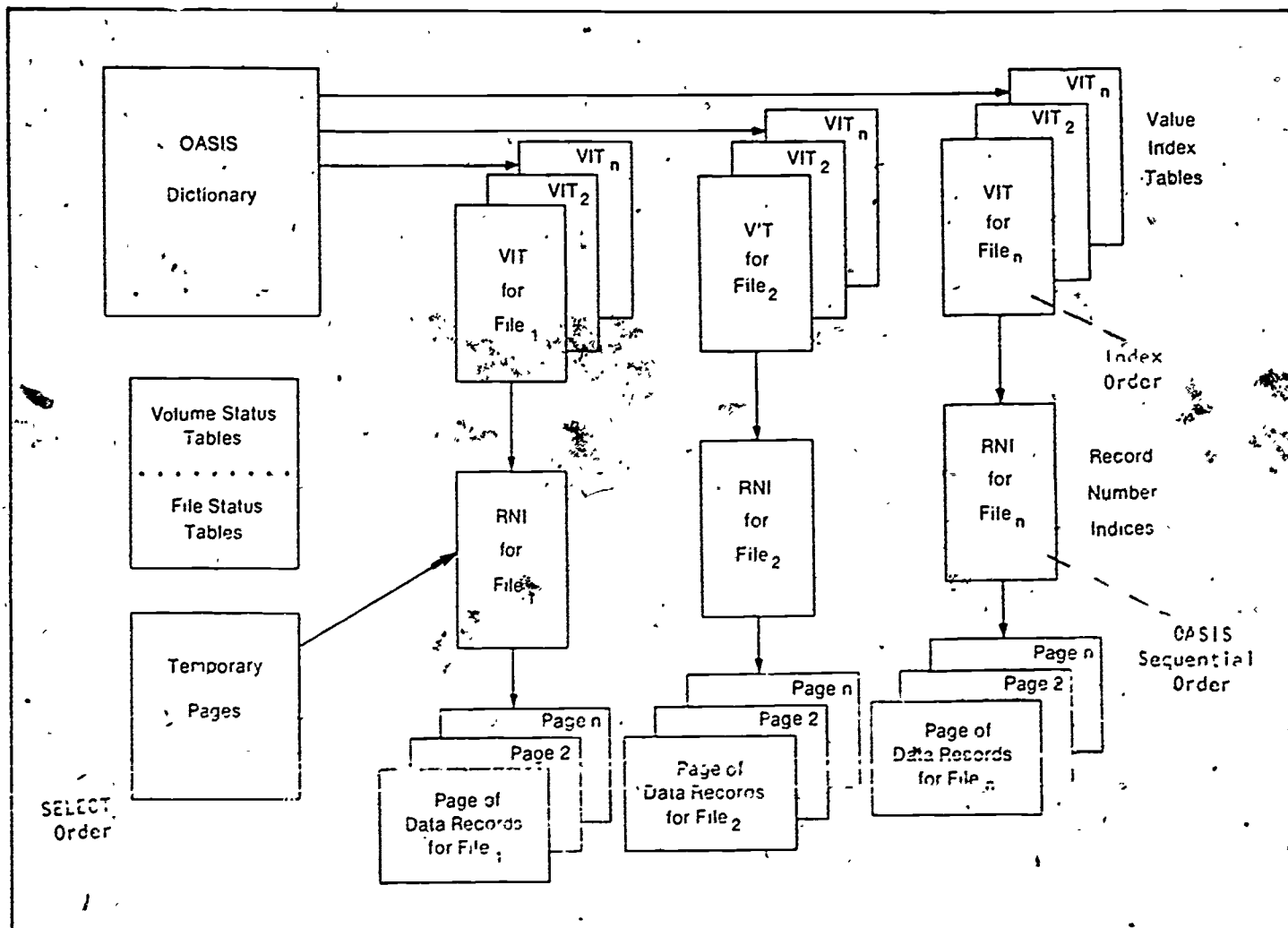


Figure 2. Oasis Data Base Components

2.2.4 Retrieval Methods. The OASIS file structure was designed specifically for several different retrieval methods. The physical layout of the data base itself is also oriented toward these methods. All of the disk space is divided into 1692-byte blocks, called "pages", and OASIS reads in an entire

page whenever data from that page are requested. The utility of this design characteristic may best be observed by examining the various retrieval methods.

The fastest method of accessing OASIS records involves reading the records in "RNI order" (the order the records were added to the file). An RNI page is accessed to locate the physical address (page plus displacement onto that page) of the desired record and that page is then read. OASIS will average a I/O for every $(1670/\text{average-record-length})$ records (i.e., one I/O for every 33 50-byte records, two I/Os for a 3000-byte record).

Records may also be retrieved according to the index order of a given element or according to a "given value" of such an element. In this method a VIT page is read to obtain the RNI for the first record containing the first value (or the requested value). Processing then proceeds as for OASIS sequential retrieval. As in the case of RNI pages, an actual read of a VIT page is not always necessary in that as many as 540 instances of symbolic pointers may be found on one page.

The third and most flexible method of accessing OASIS records is used to "select records" into a subset of a file or in a given order other than that of a single indexed element. This method involves the use of the SELECT service. Using as input an English-like syntactical arrangement called a WHERE-clause, SELECT constructs a list of record numbers (and, optionally, accepted segment occurrences) of all records in the file which meet the specified criteria. The specification may contain as many as 20 conditional expressions connected by OR, AND, or TAND (a special connector permitting comparison of data within a segment occurrence) and up to ten ascending and/or

descending sort fields. Parentheses are supported, and operators include EQ, NE, GR, LS, LE, GE, and RN (range).

The power of the facility is derived from the fact that any element may be made part of the WHERE-clause. During the parse of the selection criteria, SELECT first utilizes the indexed elements to form a set of RNIs satisfying the criteria specified for the indexed items. If necessary, those records are then accessed to determine the validity of unindexed data. The output of SELECT is usually a list of RNIs (which may be saved and reused at a later time if desired). Thus, the entire record is available for access, without any duplicate storage space for temporary files, regardless of the elements in the selection criteria. It is also possible to perform selection by segment (thus, element) occurrence in such a manner that the caller is returned only those segment occurrences within the record that satisfied the selection criteria.

The various methods of record retrieval were obviously designed with different purposes in mind. The OASIS sequential and indexed order methods facilitate rapid manipulation of large quantities of data. The physical proximity of all data for a record minimizes the time needed to gather all data on a record or to maintain large portions of those data at one time. Requests for a given value of a particular element are useful in selective batch maintenance as well as in online retrieval of records by key. The capabilities of SELECT for the complex reordering and subsetting of data are ideal for batch reporting manipulations and for complex online correlations of data for management decisions and also permit more simple application programming (since conditional selection need not be written into each program).

2.3 The Generalized Services. Relying heavily on the OASIS File Services capabilities, the Generalized Services provide automatic facilities by which the user may satisfy unanticipated requests immediately and produce countless varieties of reports without costly application programming (and long time delays). Both QUERY and the Terminal Report Generator provide automatic editing and make use of the full SELECT capabilities for population selection and record ordering. Full security checking is rendered at all times. In addition to displaying elements, groups, and indirect references, facilities provide for the calculation of functions (SUM, COUNT, AVG, MIN, and MAX) and arithmetic expressions (elements, functions, or literals separated by the operators +, -, /, *).

QUERY supports both descriptive and WHERE-clause requests. Descriptive queries permit the user to browse the Dictionary; to review components of a file or group, values of an element, or names of the files to which he has access. The more common WHERE-clause query is used for displaying record contents in either vertical or columnar format and may include a population count. Full paging capabilities as well as hard copy options, are provided. Figures 3 and 4 show sample WHERE-clause QUERY requests.

BASIC STUO.INFO.WHERE MAJOR RN '040'/'048' AND SEX.CODE EQ 'F'.					
MEMBER.ID	MEMBER.NAME	S M MAJ.OESC	LQR	CLS	DATE.LAS
0-1/67-024407	CORNELIUS, JODY ANOREA	F S ENGLISH	3/69	JR	07/26/71
0-1/67-043601	HENKE, SUZETTE ANN	F S ENGLISH	4/69	GRAO	07/23/71
0-4/68-065612	MINCHENBERG, NANCY LEE	F S ENGLISH	3/69	JR	07/26/71
0-1/67-005373	MYDANS, SHELLEY	F S ENGLISH	3/69	SR	07/26/71
0-1/66-060043	FERRARI, TERESA MARY	F S HISTORY	1/69	SR	07/26/71
0-1/69-055034	GILBERT, DEBORAH	F S HISTORY	3/69	SR	07/22/71
0-1/69-034426	MILLS, ANNETTE M	F M HISTORY	3/69	JR	07/26/71
PAGE 1					
PAGE? ('ENO', 'N', 'P', 'L', X): N					

First Page of QUERY Showing Information on Female Students in Several Departments

Figure 3

MEMBER.ID COURSE/INFO WHERE DEPT.ABBR EQ 'ENGL' TAND CREDIT.UNITS EQ 5 TAND GRADE EQ 'A'											
MEMBER.ID	DEPT	DEP	CRS	S	SC	COURSE.TITLE	CR	GR	QYY		
0-1/69-014403	ENGL	400	002	0	01	FRESHMAN	05	A	369		
	MATH	450	043	0	01	ANAL GEOM CALC	05	B+	369		
	PHYS	570	053	0	01	ELECTRICITY	04	B	369		
	PHYS	570	054	0	01	ELECTRICITY LAB	01	B+	369		
	FR S	992	001	B	01	VEHICLE DYNAM	03	A+	369		
0-1/67-014902	W PE	092	016	0	06	INT TENNIS	01	A	369		
	ENGL	400	005	0	01	NARRATION	03	B	369		
	HUM	440	005	0	01	HUMANITIES SEM	05	A	369		
	ENGL	400	200	0	02	AMERICAN LIT	05	A	369		
	CS	670	005	0	01	COMPUT PROGRAM	03	+	369		
PAGE 1											
PAGE7 ('END','N','P','L',X): 3											

QUERY Showing Academic Program of Students with a Particular Course and Grade

Figure 4

The OASIS Terminal Report Generator (TRG) was designed as a vehicle by which the programming of a large portion of report-oriented applications might be avoided yet was based on the assumption that no generalized report definition vehicle could provide for every complicated form of report in an efficient and user-oriented manner. Report definition is accomplished in an interactive mode with the user answering such terminal-prompted questions as desired report name and output mode. (Definition may also be performed in batch mode.) Automatic page numbering and/or dating may be specified. Elements may be designated as control breaks so that total and summary functions will be performed when the value of any of these elements changes.

As the format and content of each line (and associated headings, if any) are defined, the user can modify the automatic edit pictures and intersperse text. Both data lines (for the detail data of the report) and total lines (for production whenever one of the predefined control breaks changes) may be included.

When the definition is complete, OASIS generates object code for that report and stores it permanently in a special file. The user may then produce the report or request a proof based on a subset of data. The report may be displayed on the terminal (with paging capabilities similar to those used in QUERY), spooled from the terminal onto the OASIS log tape for later production in batch mode, or initiated directly from the batch partition.

2.4 Tailored Services. Although QUERY and TRG were designed to eliminate many of the programming needs in an administrative environment, it was felt that for the many routine functions that would be initiated countless numbers of times per day at a terminal, the user should be required to enter the minimum of data necessary to obtain his required output - in a fashion similar to pulling a tab card out of a manual file. To facilitate the creation of such tailored services (called networks), OASIS includes support for high-level language communication with the OASIS File Services and Terminal Services. To make the programming process more convenient, online debugging facilities permit the programmer to trace execution, to show registers or selected portions of memory, and to modify (patch) register contents and selected memory locations.

The File Services routines are also available to batch programs written in Assembler language or any high-level language which supports a standard CALL statement. Such programs may be written in any style natural to the programmer and may utilize non-OASIS files and peripherals.

2.5 Summary. OASIS could be called a compromise system. It was not designed to be all things for all environments and therefore has avoided

the inefficiencies of generality. On the other hand, it was not designed with such a specific environment in mind that its flexibility and diversity would be severely limited. OASIS is an attempt to relieve the "programmer" of some of his burden, to provide the "user" with flexible means of interacting with his data, and to provide "management" with a system which can answer its needs and yet not overtax its computing resources.

THE DARTMOUTH OSCAR SYSTEM

John S. McGeachie
Director
Data Processing
Dartmouth College

THE DARTMOUTH OSCAR SYSTEM
John S. McGeachie, Director
Dartmouth College, Data Processing Center
Hanover, New Hampshire 03755

The Dartmouth OSCAR (On-line Student Course Assignment and Registration) system provides the Office of the Registrar with an on-line capability for assigning students to courses, updating course offerings and distributing students among course sections. This paper describes the use of OSCAR during the course assignment process for a typical term, a brief overview of the OSCAR system and the procedures for changing student course selections and updating course offerings.

1. INTRODUCTION

The OSCAR system operates on the Dartmouth Time-Sharing System (DTSS) [Ref. 1,2,3] and is accessed through terminals located in the Registrar's Office. During the busiest periods of each quarter, the Registrar and his staff typically use about five terminals; DTSS normally supports about 150. Approximately 3850 of Dartmouth's 5400 students are actively involved in the course assignment and changing process each quarter, and the normal academic load is three courses per student.

The predecessor to the OSCAR system required a great deal of manual work. As the number of course offerings increased, as students began shuttling back and forth between on-campus and off-campus programs, and as the restrictions on course changes became less stringent, the manual and punched card system began to show signs of severe stress. In response to these pressures the OSCAR system was developed during the winter of 1972. Both systems ran in parallel during the spring, and the OSCAR system was put into full operation during the summer (the lightest term of the year).

The following sections describe (i) OSCAR's use during the course assignment process for a typical term, (ii) an introduction to the OSCAR system; (iii) student course selection changes; (iv) course offering changes; and (v) miscellaneous features of the OSCAR system.

2. CHRONOLOGY OF REGISTRATION EVENTS

Several months prior to the start of a new term, course selection cards (called "elective" cards) are sent to students. At the same time, an on-line file of tentative course offerings is created. This file contains an entry for each course, but does not include course sections or laboratory periods. Special restrictions required by a department are included, however. For example, certain courses are open only to upperclassmen, others have an enrollment limit, or require instructor permission, etc. When the elective cards are returned by the students (usually about six weeks before the start of the term), they are fed into the OSCAR system and used to develop a preliminary assignment of students to courses. Restrictions and permission are automatically checked by the system at this time.

Following the preliminary course assignments, enrollment summaries are sent to each academic department. The departments use these summaries to determine a) the number of sections required for each course, including laboratory and discussion periods if appropriate, b) the times at which the various sections will be offered, and c) the section instructors. For example, a Freshman English course might have as many as twenty sections; an elementary chemistry course might require anywhere from two to ten laboratory periods.

The data received from the academic departments are used to create a new OSCAR course data base, containing courses, sections, laboratory and discussion group periods together with the corresponding instructors and time periods.

A revised assignment of students to courses is now done using the new course data base. This process is called sectioning and attempts to assign students to courses and sections in such a way as to avoid student time conflicts and keep a balanced enrollment among the various sections for each course. Avoidance of time conflicts takes precedence over balanced enrollment. In case of an unresolvable conflict, a list is printed showing the student's name, his requested courses, the particular courses, sections and times available, and the reason for the conflict. The Registrar's office later examines this list and takes appropriate action on each individual case. While the algorithm used is relatively simple - for example, no attempt is made to automatically reshuffle previously determined course assignments - it is an enormous improvement over the previous manual methods. At the completion of this process class lists for each course are mailed to the appropriate instructors.

After registration, there is a five to ten day grace period during which students may switch courses. When a student requests a course change, a staff member of the Registrar's office, sitting at a terminal, enters the student's identification number, withdraws him or her from the course to be dropped, and specifies the new course requested. If the desired course requires special permissions, the system will ask the terminal operator if the student has the required authorization. If the course has sections, the system will attempt

to utilize the section with the lowest enrollment and which does not have a time conflict with the student's other courses. If the course has required laboratory or discussion periods, the student will be placed in one of them, using a procedure similar to that used for course sections. At the completion of the process, OSCAR notifies the operator of the particular assignment made, and the student is provided with immediate feedback concerning the success of his request. During the first few days of the term, as many as five terminals may be connected to the course changing program, which continuously updates the course data base. At the end of each day, a list of "add/drop" notices are prepared for each course and mailed to the respective instructors. These lists identify students who have been dropped from or added to each course, thus permitting the instructor to maintain an up-to-date course enrollment list.

3. THE OSCAR SYSTEM

The most important programs in the OSCAR system are:

CHANGE	Used to update student course assignments;
SECTION	Used once per term to do sectioning;
UPDATE	Used to add new courses or make major revisions in student data.

In addition, there are a number of initialization and report generation programs which are not worth describing here.

3.1 Accessing CHANGE. CHANGE is a "multiple-terminal" program [4] which can be used by several people simultaneously. The first person to begin a session with CHANGE is known as the "master" user, and must select a one to

eight-letter "keyword" (e.g., brownfox, syzygy, or whatever). This user then signs on to the time-sharing system under the user number corresponding to the desired term (there is a separate user number for each of the Fall, Winter, Spring and Summer terms). The master user then types

OLD REGLIB***:CHANGE

After the computer responds "READY", the master user types

LINK keyword, *

to initiate the "conference" and to indicate that any number of additional terminals (up to a maximum of nine) may join the conference at any time.

Users on other terminals may do this by simply typing

JOIN keyword

where the keyword used in the JOIN command must be the same as the one used in the LINK command. There is nothing special about the "master" user; it is simply that one person must initiate the conference, by calling up the program and typing the LINK command, and he is called the master user. The joiners may be signed in under any user number and do not have to type OLD before typing JOIN.

After a user has entered the conference (either by initiating it or joining it), the CHANGE program will type the message

ENTER INITIALS, PASSWORD?

to which the terminal operator responds with his or her initials and the appropriate password.

The password provides authorization: it would be undesirable to permit students to juggle their own, and perhaps their unwitting classmates' (?)

course schedules. The date, time and operator's initials are written to an audit trail file, which also records every student-related transaction for later use by various reporting programs.

A brief summary of the more important commands available in CHANGE is provided below:

ADD	Add a student to a course
DROP	Drop a student from a course
LIST	List a student's courses
FEE	Charge a student a fee
CLIST	Print a class list at the terminal
STATUS	Print the status of a course (enrollment, etc.)
EXIT	Terminate a session with CHANGE
FIND	Find student's id. number given his last name
PASSWORD	Change initials and password level
WHO	Provide a list of the other conference participants

3.2 The SECTION program. The SECTION program is used for pre-registration sectioning of the entire student body. It is similar to CHANGE, but has a more restricted command list, may be used by only one terminal at a time, and accepts commands from a file (as well as from the terminal). SECTION is accessed in the same manner as any other program on DTSS: the terminal

operator types "OLD REGLIB***:SECTION"; when the system answers "READY"; the

operator types "RUN" [5]. The principal commands are:

ADD	Add a student to a course
DROP	Drop a student from a course
EXIT	Terminate the SECTION program
SECTION	Specify the name of the file of commands

3.3 The UPDATE program. The UPDATE program is used primarily to offer additional courses or change the parameters on existing courses. The

primary commands are:

ALTER	Alter the parameters of a course
OFFER	Add a new course to the term's offerings
WITHDRAW	Delete a course from the current term's offering ..
REASSIGN	Generate a file of ADD commands for SECTION
EXIT	Terminate the UPDATE program
STATUS	Print the status of a course (enrollment, etc.)

3.4 Passwords. When an OSCAR program requests a password the user may enter one of three passwords. The first password grants access to commands that enable the user to examine student course assignments, but not to change them (CLIST, EXIT, LIST, STATUS, WHO). The second password grants access to commands that enable the user both to examine and update student course assignments (ADD, DROP, FEE, REASSIGN, SECTION). The third password grants access to all of the commands available in any one program, including the ability to alter the fundamental parameters of a course and to make available new courses (ALTER, OFFER, WITHDRAW). Thus personnel outside the Registrar's office (e.g., in the Dean's office) may be given permission to list a student's courses but not to change them.

4. CHANGING STUDENT COURSE ASSIGNMENTS

The primary vehicle for changing student course assignments is the CHANGE program, although the SECTION program is also used for this purpose during sectioning. The ADD command, which assigns a student to a course, is the most complex and is therefore described in considerable detail. Brief descriptions are provided for the other commands. The PASSWORD and WHO commands

are omitted. Appendix A contains an example of a dialog with the CHANGE program.

4.1 The ADD command. The ADD command will enroll a student in a course and update its enrollment count. If a course is sectioned, the student will be placed in the section with the smallest enrollment which does not have a time conflict with his other courses. If a course has laboratory or discussion groups, the student will be placed in the smallest of these which does not have a time conflict with his other courses.

To specify a particular course section, laboratory or discussion group the course name may be followed by the letters S, L, or D and the number of the desired section, as in "ENGL 5 S 1" or "CHEM 51 L 2" (spaces are not significant). Some examples are shown below:

ADD 107235, ENVS 35, ART 2, ENGL 5	
ADD REL 46, PSYC 1	
ADD PHIL 1 S3	(specify section 3)
ADD BIOL 4 L1	(specify lab section 1)
ADD GOVT 5 D6	(specify discussion section 6)
ADD GOVT 5, GOVT 5D6	(enroll student in GOVT 5, and also in discussion section 6)

Note that section may be abbreviated "SEC" or "S" laboratory may be abbreviated as "LAB" or "L" and discussion group may be abbreviated as "DIS" or "D". The student id. number may be omitted if a previous command (ADD, DROP, LIST, FIND) referred to the same student.

The program has a number of built-in checks for courses with enrollment limits, permission requirements, and so on:

4.1.1 Course closed

If a course is closed, the message

course name IS CLOSED
DO YOU WISH TO ENROLL STUDENT ANYWAY (YES OR NO)?

will be printed at the terminal. If the answer is YES, the student will be enrolled in the specified course. If NO, the student will not be enrolled in the course.

4.1.2 Course full

If a course is full, the message .

course name IS FULL
DO YOU WISH TO ENROLL STUDENT ANYWAY (YES OR NO)?

will be printed at the terminal. For this and all other questions, the operator may answer YES or NO as in 4.1.1 above.

4.1.3 Course requires special permission

If a course requires a special permission, the message

CHECK SPECIAL PERMISSION LIST FOR THIS COURSE
DOES student name HAVE IT (YES OR NO)?

will be printed at the terminal.

4.1.4 Course closed to freshmen

If a course is closed to freshmen, the message

course name IS CLOSED TO FRESHMEN
DO YOU WISH TO ENROLL STUDENT ANYWAY (YES OR NO)?

will be printed at the terminal.

4.1.5 Course closed to upper-classmen

If a course is closed to upper-classmen, the message

course name IS CLOSED TO UPPER-CLASSMEN
DO YOU WISH TO ENROLL STUDENT ANYWAY (YES OR NO)?

will be printed at the terminal.

4.1.6 Course open only to seniors

If a course is open only to seniors, the message

```
course name IS OPEN ONLY TO SENIORS  
DO YOU WISH TO ENROLL STUDENT ANYWAY (YES OR NO)?
```

will be printed at the terminal.

4.1.7 Course requires instructor permission

If a course requires instructor permission, the message

```
course name NEEDS INSTRUCTOR PERMISSION  
DOES student name HAVE IT (YES OR NO)?
```

will be printed at the terminal.

4.1.8 Time conflicts

If a time conflict is found between student's current courses and the one to which the operator is attempting to add him, the following messages will be printed:

```
STUDENT student name IS ENROLLED IN  
(... list of student's courses...)
```

```
THERE IS A TIME CONFLICT BETWEEN STUDENT'S CURRENT COURSES AND
```

```
(... course with time conflict...)
```

```
DO YOU WANT TO ENROLL STUDENT ANYWAY (YES OR NO)?
```

If the operator answers YES, the student will be enrolled in the course even though it has a time conflict with his other courses.

The ADD command will accept a series of trailing letters as part of the course specifier to indicate that a student already has permission for a course.

The legal trailing letters are listed below:

"C" indicates that student is to be enrolled in course even though it may be full or closed

"P" indicates that student is to be enrolled in course even though it may require permissions or be closed to the student's class (e.g., a freshman taking a course open only to upperclassmen).

"D" indicates that a student is to be enrolled in the same course twice; a "D" will be printed to the left of a student's dual enrollment courses when these are displayed using the list command.

"X" indicates that student is to be enrolled in course even though it may require special permissions.

Examples:

ENGL 5 SEC 20 P (override course permission requirement)
 ART 2C (override closed course)
 ANTH 60 CP (override both permission and full course)

4.2 The DROP command. The DROP command will delete a student from a course, provided that he is currently enrolled in it. The student will also be dropped from the associated laboratory or discussion group, if the course has one. Some examples are given below:

DROP 175035, MATH 13
 DROP 4520, ANTH 1, PHYS 13, MATH 6
 DROP ART 21 (use current student idno)
 DROP BIOL 17 L2 (drop laboratory section 2 only)
 DROP GOVT 44 DIS 3 (drop discussion group 3 only)

4.3 The LIST command. The LIST command displays the specified student's assigned courses in the following format:

STUDENT idno name class:
 course time special indications (if any)

with one line of output for each course. An example of the command is

"LIST 40000W", which would show the course being taken by student 40000W.

4.4 The FEE command. The FEE command accepts a dollar amount to be charged to the current student for course changes. A billing record will be appended to the audit trail file for later processing. The FEE command

must have been immediately preceded by an ADD or DROP command which resulted in a course change.

There is no way to reverse changes through the system. If a fee is erroneously applied to a student, a manual correction must be issued to the Comptroller's office.

4.5 The CLIST command. The CLIST command generates a list of all the students enrolled in the specified course, laboratory or discussion group. Identification number, full name and class are printed for each student. The program will pause after listing every 15 students and the message
CONTINUE (YES OR NO)?

will be printed. If the operator answers NO, the class list will be terminated; otherwise the next group of 15 students will be listed.

Examples are:

```
CLIST BIOL 4
CLIST PHYS 13 L6
CLIST CHEM 57
```

4.6 The STATUS command. The STATUS command displays the time, enrollment, limits and permission status of a course. If the course has sections, labs or discussion groups, this information will be displayed for all the sections, labs, etc. It is also possible to request the status of a particular lab or discussion group. For sectioned courses the total course enrollment will be printed on the first line. An example is shown below:

COMMAND? STATUS ART 15

ART	15	TOTAL:	31				
ART	15 SEC	1 P	***	14 11/21/74	MTH	13:00 - 16:00	MH1
ART	15 SEC	2 P	***	17 11/21/74	MTH	13:00 - 16:00	MH1

In the preceding example, the items following the course and section number on the last line are interpreted as follows:

<u>Item</u>	<u>Interpretation</u>
P	section requires permission
***	section has no enrollment limit
17	section currently has 17 students
MTH etc.	section meets Monday and Thursday from 1 to 4.
MH1	the Registrar's coded abbreviation for the specified time period

4.7 The EXIT command. The EXIT command terminates a session with a course enrollment system program.

4.8 The FIND command. The FIND command attempts to locate the specified student's identification number for use in subsequent ADD, DROP or LIST commands. The output from this command will depend on whether or not the program was able to uniquely identify the student's name.

4.8.1 If the system was able to locate and identify the specified student, the message

STUDENT idno name class FOUND

will be printed. The student's idno becomes the "current" idno and will be used in subsequent ADD, DROP or LIST commands until the terminal operator specifies a new student identification number.

4.8.2 If the system was not able to uniquely identify the specified student, a list of names will be printed:

idno	name	class
idno	name	class

idno	name	class
------	------	-------

followed by the message

ENTER STUDENT IDNO OR 'CANCEL'?

For example:

COMMAND? FIND BUSCH

POSSIBLE MATCHES AGAINST BUSCH ARE:

120807	BUSCH ANDREA	75
120896	BUSCH MARGARET R	76

ENTER STUDENT IDNO OR 'CANCEL'? CANCEL

(The names are real but the id. numbers are false).

5. UPDATING COURSE OFFERINGS

Existing courses may be altered, new courses added or old ones withdrawn using the UPDATE program. All course alteration commands require a higher password level than student course assignment commands.

- 5.1 The ALTER command. The ALTER command provides the ability to change the division, instructor, enrollment limit, permissions, time and title of a course. Although it is also possible to modify the enrollment, there should never be a need to do this as the enrollment total is automatically updated when students are added to or dropped from a course. The elements of a course are called fields, and each field is denoted by a three-letter abbreviation as follows:

DIV division code
ENR enrollment
INS instructor number
LIM enrollment limit
PER permissions
TIM time code
TTL course title

5.1.1 DIV: the division code is a number between 0 and 9 and indicates type of divisional credit to be received by a student taking the course.

5.1.2 ENR, LIM: these fields accept a three-digit number giving the new limit or the new enrollment count.

5.1.3 INS: the instructor number is currently the Registrar's three-digit identifier, although provision exists for eventual use of the full employee number.

5.1.4 PER: the arguments to the permission field may be one or more of the following:

"C" (course is closed)

"P" (course requires instructor permission)

"X" (course has special permission)

"F" (course is closed to Freshmen)

"U" (course is closed to Upperclassmen)

"S" (course is open only to Seniors)

The new permissions replace existing permissions on a course.

5.1.5 TIM: the new time is specified by one of the standard three-character time codes used by the Registrar's office.

5.1.6 TTL: the title must be enclosed in quotation marks (") and may not exceed 20 characters, as in TTL "CONSULT FR REGISTRAR".

Only those fields specified with the ALTER command will be updated; all other fields remain the same.

Examples:

ALTER MATH 13, TIM MT3, PER PFX, TTL "LINEAR ALGEBRA"
ALTER ANTH 6, TIM 10, INS 357, LIM 100

5.2 The OFFER command. The OFFER command allows a new course to be added to the list of course offerings. The format of the command and its arguments is exactly the same as that of the ALTER command.

A time code and title must be specified for a new course, and the command will be rejected unless they are provided. (If the instructor number is omitted, the default number 000 will be provided. This will result in the instructor name "THE STAFF" in all course list printouts.)

If a course has no sections, and a section is added, the new section must be numbered 2 or above. The main course will become section 1.

The rules for new LAB/DIS units is as follows:

5.2.1 if no previous LAB/DIS units exist, the new one must be numbered 0 or 1; in either case, the system will record it as number 0

5.2.2 if previous LAB/DIS units exist, the new one must be numbered 2 or greater; the existing LAB/DIS unit will become number 1.

Examples:

OFFER CHEM 57, INS 342, TIM 2, PER PC, LIM 50, DIV 3
OFFER ANTH 6, INS 357, TIM 3, TTL "ELEMENTARY ANTHROPOLOGY"
OFFER CHEM 57, LAB 0, INS 342, TIM 12

5.3 The WITHDRAW command. The WITHDRAW command removes a course from the list of course offerings, provided that no students are currently enrolled in it. If there are students enrolled in the course, an error message will be printed and the WITHDRAW command will not be completed.

If the course has any associated sections, laboratory or discussion groups, these will also be withdrawn provided that the enrollment is zero. It is also possible to withdraw a particular section, laboratory, or discussion group; to do this the course name must be followed by the letters S, L or D and the number of the desired section, as in "GOVT 6 D5".

Examples:

```
WITHDRAW CHEM 57      (withdraw entire course)
WITHDRAW BIOL 17 L2   (withdraw laboratory section 2 only).
WITHDRAW GOVT 44 S4   (withdraw section 4 only)
```

5.4 The REASSIGN command. The REASSIGN command is intended for use in expanding or contracting the number of sections in a course, and generates a file containing a list of DROP commands for all students in the particular course, followed by a list of ADD commands for all students in that course. This file is then used as input to the sectioning program, which redistributes the students among the revised number of sections.

The format of the output file is:

```
DROP      idno, course
DROP      idno, course
.
.
.
ADD       idno, course
ADD       idno, course
.
.
.
```

Example:

```
REASSIGN ENGL 5
```

5. THE AUDIT TRAIL

An audit trail is kept of all transactions which affect a student's course enrollment. In addition, an entry is made in the audit trail whenever one of the main OSCAR programs (CHANGE, SECTION, UPDATE) is run, and an entry is made whenever a user accesses the OSCAR system. Thus the audit trail shows the program and the user involved in all course changes. Several reports are produced from the audit trail files: (i) a report showing all changes made by each student; (ii) a report showing the daily changes in the enrollment of each course (used to update faculty class lists); and (iii) a report in chronological order showing all course changes made during a given term. This latter report has been very useful to the Data Processing staff in tracing some obscure program bugs which have been encountered from time to time.

There are four basic types of records in the audit trail files;

- 6.1 Program entry records. These show the program identification (CH for CHANGE, UP for UPDATE, etc.), the date and the time.
- 6.2 User entry/exit records. These records are created whenever a user enters or departs from an OSCAR program. These records contain the program identification, the date, the time, and the user identification (three initials).
- 6.3 Course addition/deletion records. These records contain the user identification, the date, the time, the student number, a list of courses, and an indication of whether the courses were added or dropped.
- 6.4 Fee records. These records contain the user initials, date, time, student identification number and the dollar value of the fee.

(Underlined entries are typed by terminal operator)

join RAIN

(join conference "RAIN")

TERM: 74F

ENTER INITIALS, PASSWORD? jsm, (supply initials, password)COMMAND? find bsuch (operator error)

BSUCH NOT FOUND

COMMAND? find busch

POSSIBLE MATCHES AGAINST BUSCH ARE:

12...8	BUSCH ANDREA	75	(all id. numbers have been altered for security)
12...3	BUSCH MARGARET R	76	

ENTER STUDENT IDNO OR 'CANCEL'? 12...3 (enter second id. number)COMMAND? list (list student's courses; note omission of id. number)

12...3	BUSCH MARGARET R	76	:	
COLT	42	MWF	10:00 - 11:05, TH	12:00 - 12:50 10 *
GRS	1	MTTHF	9:00 - 9:50, W	9:00 - 9:50 9 *
MATH	6	MWF	13:45 - 14:50, W	15:00 - 15:50 2

COMMAND? add econ 1

COURSE ECON 1 SEC 4 NEEDS INSTRUCTOR PERMISSION

DOES BUSCH MARGARET R 76 HAVE IT (YES OR NO)? no

***ADD-*NOT* COMPLETED.

COMMAND? status econ 1

ECON	1	TOTAL:	215	
ECON	1 SEC	1 P	45	43 10/23/74
		MWF	10:00 - 11:05, TH	12:00 - 12:50 10
ECON	1 SEC	2 P	45	(42) 10/02/74
		MWF	12:30 - 13:35, W	16:00 - 16:50 12
ECON	1 SEC	3 P	45	49 10/07/74
		MTTHF	9:00 - 9:50, W	9:00 - 9:50 9
ECON	1 SEC	4 P	45	(40) 10/25/74
		MWF	11:15 - 12:20, T	12:00 - 12:50 11
ECON	1 SEC	5 P	45	(41) 11/01/74
		MWF	11:15 - 12:20, T	12:00 - 12:50 11

COMMAND?

* note time conflicts; of the remaining sections, number 4 has smallest enrollment (40) and hence was tentatively selected by the program before requesting permission check.

Vol. II

add art 15

12...3	BUSCH MARGARET R	76	IS ENROLLED IN:	
COLT 42	MWF	10:00 - 11:05, TH	12:00 - 12:50	10
GRS 1	MTTHF	9:00 - 9:50, W	9:00 - 9:50	9
MATH 6	MWF	13:45 - 14:50, W	15:00 - 15:50	2

THERE IS A CONFLICT BETWEEN STUDENT'S CURRENT COURSES AND
ALL AVAILABLE SECTIONS OF ART 15
PLEASE CHOOSE FROM AMONG THE FOLLOWING:

ART 15 SEC 1 P	***	14 11/21/74	MTH	13:00 - 16:00	MH1
ART 15 SEC 2 P	***	17 11/21/74	MTH	13:00 - 16:00	MH1

ENTER DESIRED SECTION/LAB/DISC OR 'NONE'? 1
COURSE ART 15 SEC 1 NEEDS INSTRUCTOR PERMISSION
DOES BUSCH MARGARET R 76 HAVE IT (YES OR NO)? no
***ADD *NOT* COMPLETED

COMMAND? status grs 1

GRS 1	***	(25) 12/05/74		
	MTTHF	9:00 - 9:50, W	9:00 - 9:50	9

COMMAND? drop grs 1 (drop Greek & Roman Studies 1)COMMAND? status grs 1

GRS 1	***	(24) 12/05/74	(note drop in enrollment)
	MTTHF	9:00 - 9:50, W	9:00 - 9:50 9

COMMAND? list

12...3	BUSCH MARGARET R	76		
COLT 42	MWF	10:00 - 11:05, TH	12:00 - 12:50	10
MATH 6	MWF	13:45 - 14:50, W	15:00 - 15:50	2

COMMAND? <u>add grs 1</u>	(restore course)
GRS 1	MTTHF 9:00 - 9:50, W 9:00 - 9:50 9

COMMAND?

clist art 15 sl

(class list for ART 15 SEC 1)

ART 15 SEC 1 ENROLLMENT AS OF 12/05/74

05...0	BEATTIE JAMES L	76
20...4	DANN JESSE C	75
23...0	DONOVAN MARY E	76
40...V	ALI SHIRIN	77
40...S	DELL JAMES R	77
40...N	FONTAINE ANNE E	77
40...D	GLICKMAN KENNETH P	77
40...N	JENNINGS JOHN G III	77
40...M	LANDERS JOHN C	77
40...Q	SHEMI BULENT HAMIT	77
40...Q	TOWNSEND LUCY M	77
63...6	MUSTARD MARION M	76
83...7	STANLEY ALFRED T JR	76
85...8	SULLIVAN TIMOTHY J	76

14 STUDENTS ENROLLED IN ART 15 SEC 1

COMMAND? list 40...v

40...V	ALI SHIRIN	77	:
ART	15 SEC 1	MTH	13:00 - 16:00 MH1
ART	16	TF	13:45 - 16:45, TH1...
MATH	13 SEC 2	MWF	10:00 - 11:05, TH 12:00 - 12:50 10

COMMAND? exit

YOU ARE NO LONGER CONNECTED TO CONFERENCE "RAIN".

STOP
READY

References

1. DTSS - The Dartmouth Time-Sharing System (Kiewit Computation Center, Dartmouth College, Hanover, NH, 1973)
2. Kemeny, J. G. and Kurtz, T. E. "Dartmouth time-sharing." Science Vol. 162, 11
3. Hargraves, R. F. and Stephenson, A. "Design considerations for an educational time-sharing system." Proc AFIPS SJCC 1969 Vol. 34, AFIPS Press, Montvale, NJ, 657-664.
4. McGeachie, J. S., "Multiple Terminals Under User Program Control in a Time-Sharing Environment", CACM 16, 10 (October 1973) p. 587.
5. DTSS User's Primer (Kiewit Computation Center, Dartmouth College, Hanover, NH, 1974).

VOICE OUTPUT FOR STUDENT INFORMATION INQUIRY

David T. Clements
Senior Programmer/Analyst
District Student Information Systems
Coast Community College District

VOICE OUTPUT FOR STUDENT INFORMATION INQUIRY

David T. Clements, Senior Programmer/Analyst
District Student Information Systems
Coast Community College District
Costa Mesa, California 92626

The Coast Community College district is currently implementing several Synthesized Voice Output applications in both their Computer Aided Instruction (CAI) and On-line Student Information Systems. This paper focuses mainly on the efforts expended in providing a touch-tone/voice-output facility for Administrative inquiry of the Student Information Data Base. The author's intent is to familiarize the reader with the components of the facility and to propose its employment as an extremely effective medium from both cost and user points of view.

1. INTRODUCTION

Man has always regarded his ability to speak as an endowment which elevates him from all other earthly creatures. Whether he is now ready to include the "monstrous" digital computer into his elite status is debatable. But the technology of Audio Response has grown tremendously in the recent years and it has proven to be, in many applications ranging from Credit Inquiry to CAI for the blind [3], an ideal Man/Machine Communication.

A recent research article [5] on Voice Response listed some 18 American suppliers of speech generating equipment and systems. The available hardware appears to fall into two categories which, for the sake of delineation, might be termed Traditional Audio Response and Synthesized Voice Output. The major differences between these two types is shown in figure 1. The Traditional Audio Response unit contains sophisticated storage systems which are "loaded" with pre-recorded syllables, words, phrases or perhaps full paragraphs. Each of these "sound units" is addressable. The host program sends a request for a particular unit and it is output (or more properly, "played").

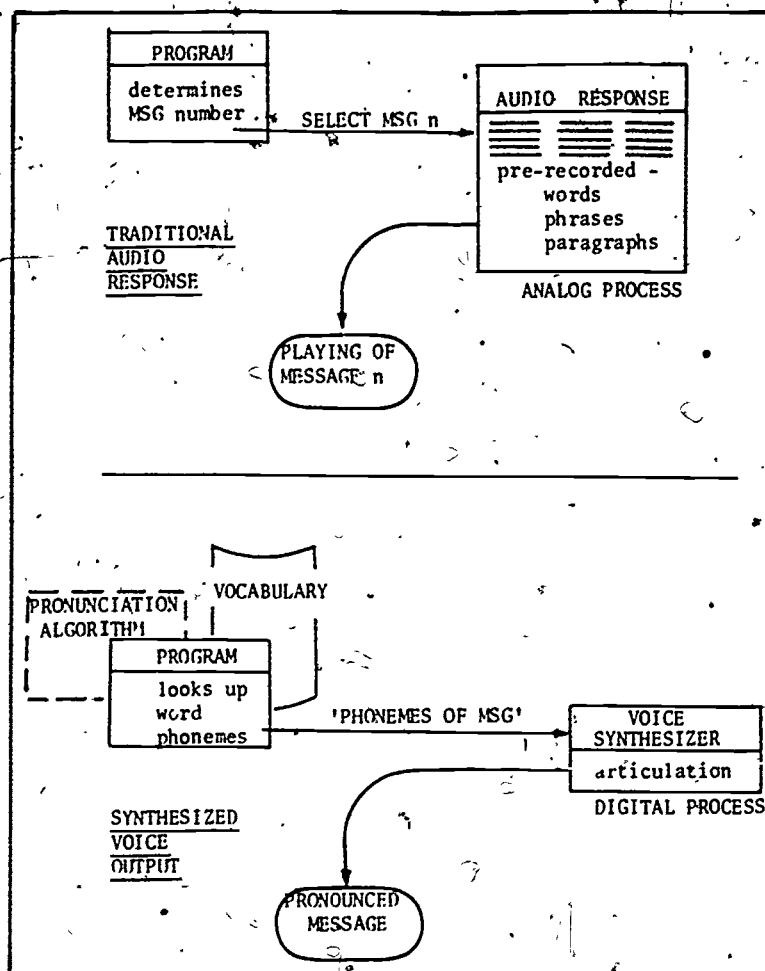


figure 1: Pre-recorded Audio Response versus Voice Synthesis

With the exception of some control unit logic, it is a total analog procedure.

A Voice Synthesizer differs from the Traditional Audio Response systems in that there are no pre-stored, pre-recorded sounds. It reacts only to digital excitement. The host program determines the phonetic structure of its output by means of a vocabulary lookup procedure or pronunciation algorithm and sends a binary string of appropriate pronunciation codes to the device for articulation.

We decided to employ a Voice Synthesizer* rather than a traditional Audio Response system for several reasons. Most notably because it seemed ideal for

*The district utilizes a VOTRAX-VITM voice synthesizer, supplied by the Vocal Interface Division of the Federal Screw Works, Detroit, Michigan.

a wide range of applications. Within our CAI systems, there are many possible uses starting with the study of speech sounds themselves on up to a full vocal interface for blind students, similar to the work in progress at Michigan State University. The Administrative side of the house also could visualize several applications which shall be summarized later in this paper.

Another attractive quality of Synthesized Voice is that there is no limit on vocabulary size such as exists with those devices requiring pre-recorded units. Vocabulary tables may be stored easily on disk. Our structure will accommodate in the area of 8000-10000 words with their phonetic codings per one 3330 disk cylinder. Also, an interactive procedure may be quickly developed to add or refine the vocabulary as it is being accessed by other tasks.

We also considered the ability to dynamically create sentences within the host program a valuable asset. That is, constant text may be spliced with variable data to create a meaningful output string.

The most convincing argument in favor of Voice Synthesis is its extreme economy. Our synthesizer may be purchased for approximately \$3500. This is well under the cost of the other Audio Response units comparable to IBM's 7770 which starts at \$60,000 and go as high as \$230,000.*

*The amounts quoted are per reference [2], specifically IBM's 7770. In all fairness, there are vendors which approach 7770 compatibility at a more favorable price. There are also several vendors supplying devices with limited vocabularies for specific application areas and their prices in some cases are significantly less. To the author's knowledge, however, there is currently only one supplier of pure Voice Synthesis equipment (as described in this paper); the Vocal Interface Division of the Federal Screw Works.

2. PRINCIPLES OF VOICE SYNTHESIZER OPERATION AND ITS REQUIRED RESOURCE

Phonetic coding for a Voice Synthesizer is shown in figure 2. Our synthesizer currently reacts to some 60 phonemes. (A phoneme is the smallest unit of speech.) There are also pausing and inflection codes. Note that the phonemes and pauses only use the low order six positions of an eight bit configuration. Inflection codes utilize the high order two bit structure.

▶ <u>THERE ARE SOME 60 PHONEMES.</u>	
PHONEME NAME	BINARY REPRESENTATION
AW (as in law)	00111101
AH (as in car)	00100100
AY (as in eight)	00100001
AE (as in cat)	00101110
B (as in book)	00001110
etc....
etc....
▶ <u>PAUSE CODES.</u>	
PAUSE NAME	BINARY REPRESENTATION
PA0 (short pause)	00000011
PA1 (med. pause)	00111110
PA2 (long pause)	00110000
▶ <u>AND INFLECTION CODES which are OR'd into bits 0 and 1</u>	
INFLECTION	BINARY REPRESENTATION
IN1 (low)	10000000
IN2 (normal)	11000000
IN3 (high)	00000000
IN4 (higher)	01000000
▶ <u>EXAMPLE:</u> The word HELP is constructed as follows-	
inflection phoneme	$\frac{3}{H}$ $\frac{1}{EH}$ $\frac{1}{UH}$ $\frac{1}{L}$ $\frac{1}{P}$
=binary	00011011 10000001 10100011 10011000 10100101

figure 2: Phonetic coding and its binary representation

An inflection is properly OR'd to a phoneme for the desired sound. The example shown for pronouncing the word HELP indicates inflection/phoneme notation and the resulting binary codes. The H sound, of course, must be stressed; thus a level 3, or high inflection. Note also that two phonemes are required to accomplish the diphthong vowel sound.

The device obviously belongs in a real-time environment. Because it has been engineered to interface per the EIA standard RS-232, there are several configuration opportunities. We have supported two different connections. Figure 3 depicts the synthesizer connected in parallel with a CAI terminal. This is accomplished by special strapping and subsetting the RS-232 interface [2]. What appears on the terminal is also spoken; an ideal coordination for trying new words and inflections.

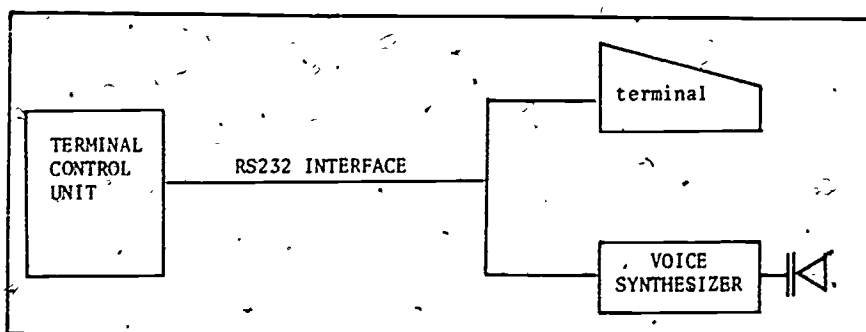


figure 3: Voice synthesizer is parallel with terminal

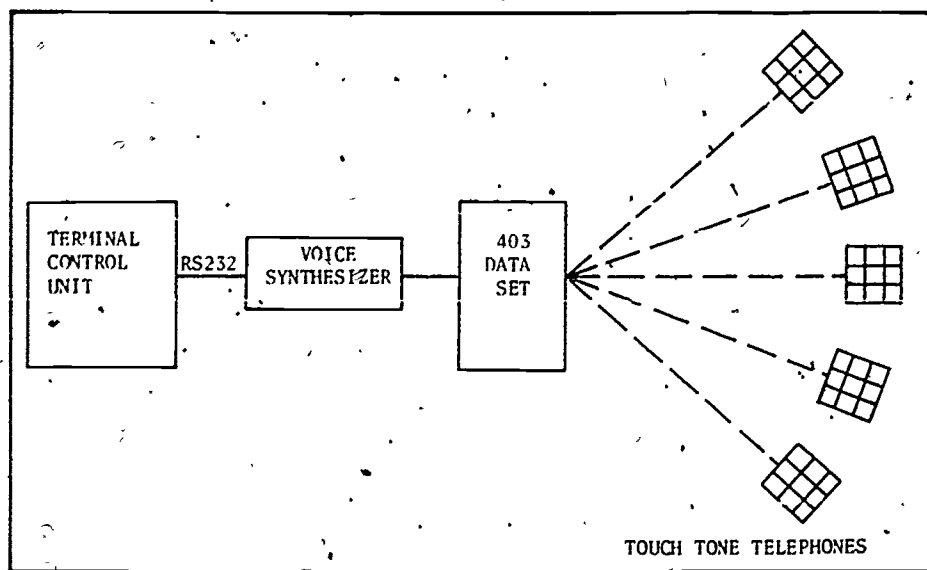


figure 4: Voice synthesizer with touch-tone interface

Our other configuration shown in figure 4 interfaces the synthesizer to a Bell 403LG data set and subsequent acoustical touch-tone interaction.*

*Unfortunately only one touch-tone telephone may interact with the configuration at a time. We are hopeful this limitation will be soon alleviated by an economic multiplexing option.

3. SOFTWARE PROCEDURES FOR CONTROLLING VOICE SYNTHESIZERS

The Coast Community College district supports two teleprocessing systems; namely, APL/SV for CAI activities and ENVIRON/1 for Administrative on-line applications.* Speech synthesis is used in both systems (ref. figure 5). Currently, we develop new vocabulary in an interactive mode under APL/SV. The vocabulary table (Lexicon) is stored in a sequential file. Of course, small vocabularies can also be stored as direct variables in the APL workspace. The APL vocabulary table is subsequently read by a batch PL/1 program which hashes the elements into a new structure called PHAST (Phonetic Hash Table). Administrative on-line applications may then access the vocabulary in ENVIRON/1 thru a common Lookup Procedure module.

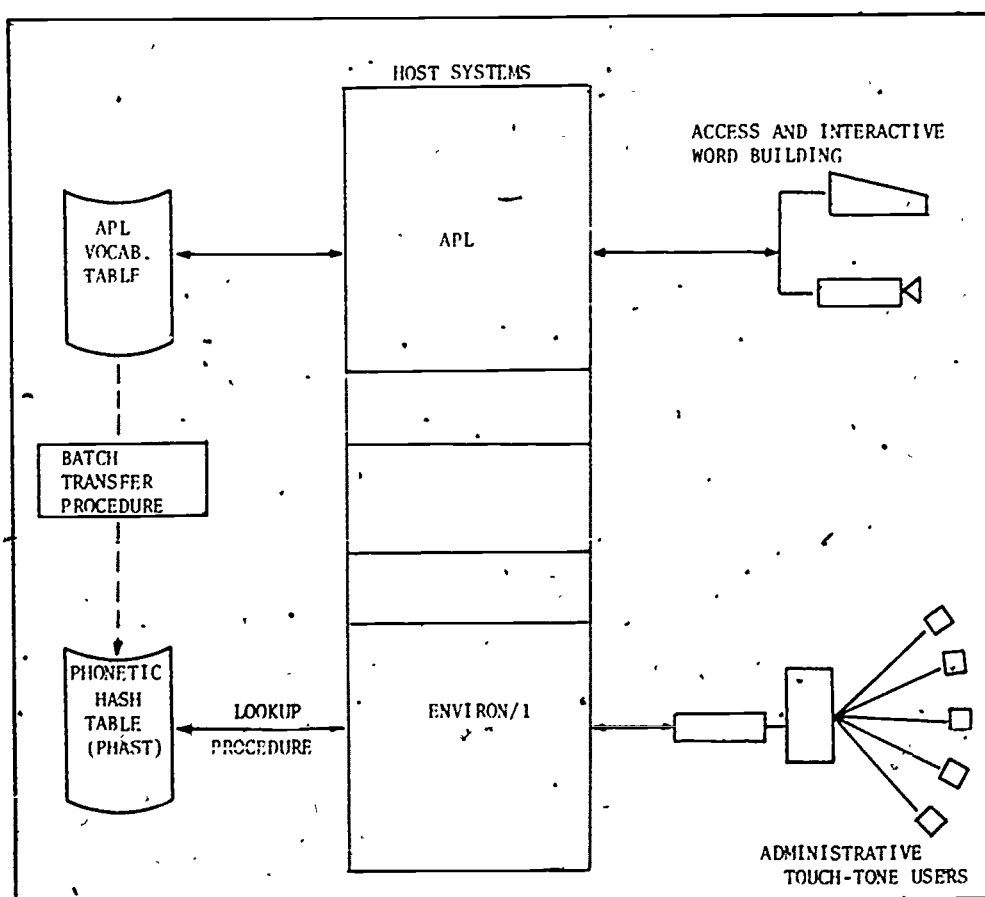


figure 5: Host systems and Voice output configurations

*APL/SV is a proprietary program product obtainable from IBM. ENVIRON/1 is a proprietary program obtainable from CINCOM Systems, Inc.

It should be noted that this scheme (figure 5) is utilized mainly for its convenience. It is not necessary to have the word-building capability singularly in APL/SV. However, the terminal-synthesizer configuration employed by APL makes the effort quite simple. You try a word, hear it, modify the phonemes and try again.

The PHAST vocabulary table is structured with a prime hash algorithm. This is an expedient organization for the random access imposed by the Lookup Procedure. The table is maintained at a load factor below 70% to enhance search performance. Our average search is accommodated by 1.8 probes. And, because of the nature of ENVIRON/1, searches for common words are often resolved in buffer store.

The rather traditional process of the Lookup Procedure is shown in the flowchart of figure 6. The application program invokes the procedure by means

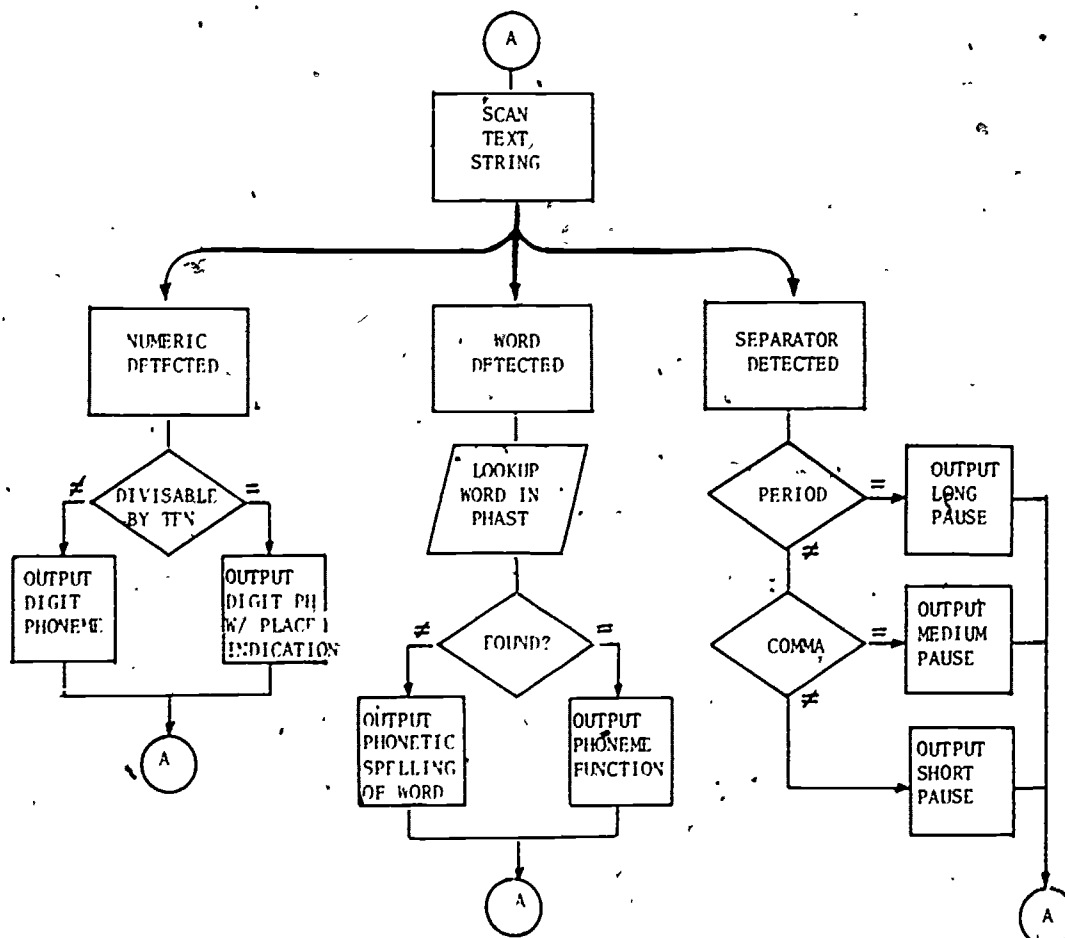


figure 6: Logical flowchart of PHAST Lookup Procedure

of a CALL and a text/sentence string parameter. Scanning determines the text elements. If a numeric is detected, it is tested to see if it is evenly divisible by 10 (modulo-10). If this is the case, the leading digit's phonemes are concatenated with the phonemes of its place indication. For instance, the numeric text 700 would be pronounced "seven" "hundred".

If an application program desires place indication verbage in its output, it purposely uses this logic. For example, if the value 1,234 is sent to the Lookup Procedure as 1000 200 30 4, it would be pronounced as "one thousand" "two hundred" "thirty" and then "four". Numeric text which is not modulo-10 is pronounced digit by digit.

When a "word" or more properly a string of contiguous characters bound by separators, is detected, it is used as a search argument into the PHAST vocabulary. If the search is successful, the associated function of phonemes is output. Otherwise, the word is pronounced, or spelled, character by character.

Separators are used to issue natural pausing. As you can see (figure 6), the presence of a period effects a long pause, a comma obtains a medium pause, and all others such as hyphen, colon, blank, etc. result in a short pause.

Mention should be made at this point about an automatic pronunciation algorithm. Bell Laboratories has published accounts of a program which produces Synthetic English Speech by Rule [4]. Although the speech produced is not inflected, it is intelligible on at least 97% of running text.

Implemented on a PDP11/45, it requires about 15,500 bytes. It contains some 750 pronunciation rules and requires a lexicon structure for words which are determined to be exceptions to the rule. This is a promising effort, and we believe our next procedural step will follow its guidelines closely.

4. VOICE SYNTHESIS APPLICATIONS FOR STUDENT INFORMATION SYSTEMS

4.1 Application Design Considerations. There are several considerations which should be included in the design of any application programs which intend to use Synthesized Voice Output. An anxiety which pops into every Administrator's mind is that the "whole world" will be able to listen in to spoken output by simply lifting up their telephone. Security, therefore, should perhaps be considered before anything else. We have two levels of security checking. The first clearance is obtained after the initial connection when the User is asked "WHAT IS YOUR USER NUMBER?" He must respond with a pre-assigned number followed by a password string of digits. If his entry at this point cannot be found within our User Security Registration list, the User is asked to input the number again. If the second attempt also fails, we hang-up the connection and write a message to the computer operator indicating that an unsuccessful "sign-on" was attempted.

Assuming a good sign-on, the User is then asked to "ENTER PROGRAM NUMBER". He, of course, must know the particular number associated with the application program. After this number is entered, the User's Security Registration record is reviewed to ascertain whether he is authorized to proceed.

Two other considerations are Brevity and Clarity. The old expression "keep it short and to the point" is an ideal mandate for Voice Output messages. It will undoubtedly be the application designer's first inclination to make the voice output as "human-like" and "natural as possible. But, no one wants to hear a long-winded computer. Abbreviated, precise sentences are very effective, especially when the User will hear the sentence everytime he accesses the application. Clarity, by the way, is often enhanced with good pausing between words. (When in doubt, throw in a pause.) Further clarity is also obtained by setting the Synthesizer's voice pitch to a low bass.

The application program modules should always be both interruptable and repeatable. These are User considerations. If he has heard all of a message he needs, he should have the capability of breaking the transmission. We use the '*' key on the touch-tone pad for this function. The User may also wish to hear a message again in the case some interference occurred. Although we do not have a "repeat" function key, all applications are divided into brief data-text modules. Any module can be repeated simply by calling for it again.

Certain limitations should be realized if the touch-tone phone is used as a "terminal". All User input can only be numeric. This imposes quite a handicap on data-entry. There are schemes, however, where the User indicates with one series of digits that the next series of digits is supposed to be alphabetic and visa-versa, but these are cumbersome, to say the least.

Anticipating the User demand for an application is an important consideration because it will hopefully help determine the traffic increase on the equipment. Unfortunately, this evaluation will lead the designer into telephone queuing theories and their complicated analysis. But, a sure death for a new application is the User community's reaction, "yeah, sure it's great, if you can ever get past the busy signal".

4.2 Student Information Inquiry. Our first voice output application for Administrative Users was designed primarily for some 50 Counselors within the district. The initial draft of the facility was purposely kept simply as we knew the implementation effort would involve a tremendous education in the technical support area. The primary function of the application is to quickly provide the Counselor with student demographic data as well as summary information on his current activities.

The flowchart in figure 7 depicts the logical construction of the application program and its User interface. After passing a resident security procedure, control is passed to the application's main module which immediately identifies itself by outputting "This is the Student Information Program". A read is then issued to the User's terminal/telephone after the prompting word "Proceed" is spoken. At this point, the User has several options. If he wishes to terminate, he inputs (keys-in) '999' whereby control will be returned to the security procedure and subsequent opportunity to select a different application program will be offered to him. The other input options are either a student number specification, or an inquiry module code. A student number should be

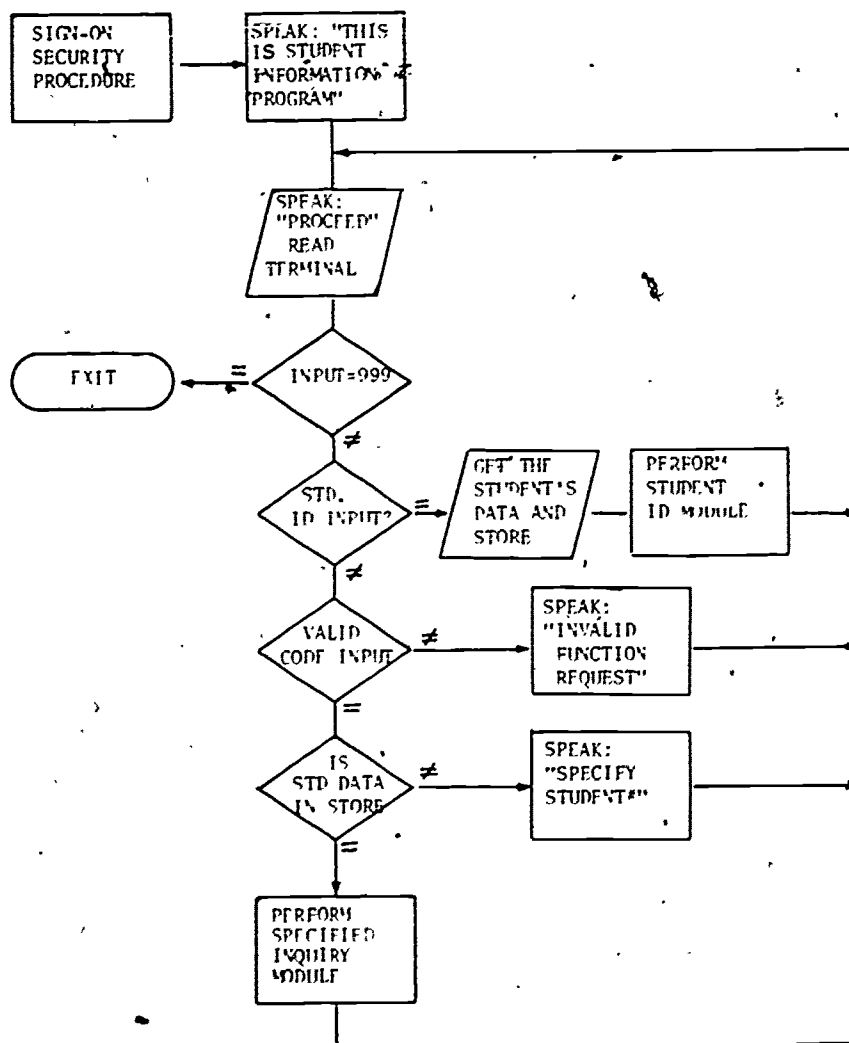


figure 7: Logical flowchart of Student Information Voice Output application program

specified first as it is the impetus for the Data Base call and the storing of the student's information for subsequent interrogation. A module which repeats the student's ID number and spells the name is performed at this point to verify that the correct student has been accessed.

Once the student's information is in storage, several brief inquiry modules may be requested, in any random order. For instance, if the User responds to "Proceed" with the digit 3 input, an associated module outputs summary information about the student's current activities/status. The text of this module is shown in figure 8. Note that variable data is easily concatenated with constant text to form one message string for the Phonetic Lookup Procedure and the Synthesizer.

<u>INQUIRY*MODULE - DIGIT 3</u>	
<u>Spoken if Withdrawn</u>	"STUDENT IS TOTALLY WITHDRAWN"
<u>Spoken if Active</u>	"STUDENT IS ACTIVELY ENROLLED IN (number) UNITS, FOR (number) CONTACT HOURS. (HE/SHE) IS WITHDRAWN FROM (number) UNITS. STUDENT IS CLASSIFIED AS A (FULL,PART) TIME (DAY, EVENING) STUDENT."
if applicable	"(HE/SHE) IS CONCURRENTLY ENROLLED IN HIGH SCHOOL."

figure 8: Output messages of inquiry module
for Student's Current Enrollment Status

We currently have eight various inquiry modules which are requested by associated inputs of 1 thru 8. The effects of these modules are summarized in figure 9. The inquiry module for the digit 2 input perhaps requires clarification. Our district is composed of two colleges. Three semesters are always maintained on-line. Hence, there are six different semester data sub-sets. The User must specify (via conversational input) which college and semester he wishes to

reference. Once specified, the reference is stored with the User's security registration, and used by all subsequent Data Base calls. The User may change the reference at anytime.

<u>INQUIRY MODULES</u>	
<u>TOUCH-TONE INPUT CODE</u>	<u>DESCRIPTION</u>
7 DIGIT ID	STUDENT'S RECORD IS RETRIEVED AND STORED. HIS ID NUMBER IS SPOKEN FOR VERIFICATION. HIS NAME IS SPELLED.
1.	A MENU OF AVAILABLE INQUIRY CODES IS SPOKEN.
2.	USER IS ASKED TO INPUT COLLEGE AND YEAR SEMESTER REFERENCE.
3.	STUDENT'S CURRENT ENROLLMENT STATUS IS SPOKEN.
4.	STUDENT'S CUMULATIVE GPA DATA IS SPOKEN.
5.	STUDENT'S MOST RECENT SEMESTER GPA DATA IS SPOKEN.
6.	STUDENT'S CHARACTERISTICS (SEX, AGE, MAJOR, ETC.) IS SPOKEN.
7.	STUDENT'S ADDRESS AND PHONE IS SPOKEN.
8.	STUDENT'S ACADEMIC STATUS (PROBATION, DEANS LIST, ETC.) IS SPOKEN.

figure 9: The various inquiry modules and their input codes

Upon completion or interruption of any module, control is returned to the prompting "Proceed" and a terminal/telephone read. (A module may be interrupted at anytime by depressing the * key.) This affords the User the ability to stop something he does not need to hear, or easily repeat a module which he could not hear well.

If the User forgets which input digits relates to the various modules, he may input a 1 which will result in a "Help" message that details a menu of the available modules within the application program. We plan to make this a standard for all future voice output applications. This is a more expedient reference than the User Manual ... (which for some reason can never be found!)

4.3 Other Applications.

As mentioned previously, we are able to visualize several other Voice Output Applications for our Administrative Users, as well as a number of extensions to the program just described.

One of our analysts is currently designing Voice Output interface to the on-line Budget System. The scope of this application will provide all department heads the capability to inquire about their particular budget accounts; the balances, P.O. status, encumbrances, etc. Because the district operates under a centralized accounting organization, this facility will expediate the often burdensome communication now in effect.

We are also laying out an application which will output numerous on-the-spot enrollment statistics as the On-line Registration procedure is in progress. This will include the capability to alter the OPEN or CLOSED status of a section.

Rather than supply each of our programmers with desk calculators, we are implementing a Voice Output Application for simple calculations. This will include the decimal, octal, or hexadecimal base specification.


There are many other ideas, which hopefully, the reader at this point can also visualize.

5. SUMMARY

The relative effectiveness of the touch-tone/voice output facility provided for our Administrative Users needs to be evaluated from two points of view; the User's and the Cost.

The User's appreciation for the facilities far outweighs his criticism. To begin with, he considers the "talking terminal" a dramatic advance in service provided by the Information Services department. He relates that he is able to react more immediately to voice than to any other means of communication.

(Immediacy, by the way, is the real key to proper voice output information processing. [5].)



The User is also pleased that he has a sufficient number of "terminals"; effectively there is one on every desk, every office. This is to say nothing of the fact that he may use his home phone, or any other off-campus instrument. And, these "terminals" are compact and easy to operate! This is to say nothing of their dependability.

The medium is only effective, however, for applications which are limited in scope. Speech rate is considerably slower than eye-scanning. Therefore, more complex and detailed output is often better communicated in printed form.

When the User first hears his talking terminal, he will complain that it does not sound right. He is correct; it has an electro-mechanical accent which takes a few minutes to become accustomed to. It is possible to achieve extremely natural sounding messages and sentences using inflections, but it is an arduous programming task and does not lend itself to the word by word lookup-output sequence. The User will quickly adjust anyway.

The most significant amount of effectiveness is realized by the cost. If, for example, a two synthesizer configuration is procured for \$8000 and they adequately support some 80 Administrative Users, the net terminal unit cost works out to be \$100 each.* And, by allowing this volume of terminals, you are really supplying an Information Service.

We felt very strongly about the possibilities of Synthesized Voice Output, and are amazed at the results we have achieved with it to date. We believe it will become a widely used medium in the near future as more Educational institutions learn of its advantages. To these installations, we of course extend our experiences and encouragement.

*Touch-tone pad adapters, ranging in price from \$60 to \$100, are available for standard telephones.

ACKNOWLEDGEMENTS

Several individuals within the district have aided the author either in the preparation of this paper, or with the implementation of the described procedures.

- Robert Schaulis (Director, district Information Services)
- John Clark, Richard Mercer and Donald Rueter (Faculty Advisors in CAI)
- Mike Benson, Systems Programmer
- Michael O'Connor, Student Assistant
- Wiley Griener, Student Assistant

REFERENCES

- [1] J.L. Flanagan, Speech Analysis Synthesize and Perception, Springer-Verlag Berlin Heidelberg, 1972, 204-276.
- [2] D.B. Rueter, Speech Synthesize Under APL, Proceedings of the Sixth International APL Users Conference, May 1974, 585-596
- [3] J.B. Eulenburg and Morteza Amir Rahimi, A Computer Terminal with Synthetic Speech Output, Behavior Research Methods & Instrumentation, 1974, Vol. 6, No. 2, 255-258.
- [4] M.D. McIlroy, Synthetic English Speech by Rule, Bell Telephone Laboratories, March, 1974.
- [5] All About Voice Response, Datapro 70, September, 1974.

FIND: A FLEXIBLE MANAGEMENT
INFORMATION SYSTEM

David A. Chapin
Manager
Project FIND, Data Processing
Dartmouth College

FIND - A FLEXIBLE MANAGEMENT INFORMATION SYSTEM
David A. Chapin, Project Manager
Dartmouth College, Data Processing Center
Hanover, New Hampshire 03755

The major emphasis during the early stages of the project has been the development of the FIND system as an easy-to-use information retrieval tool. The "host language" contains simple, yet powerful, commands for data retrieval, query, maintenance, and display. The command structure of the FIND language utilizes a number of active verbs to describe the type of operation to be performed on a working data set: RETRIEVE, SELECT, SORT, PRINT, UPDATE, RESTORE, etc. These commands are frequently modified by argument lists indicating the particular scope a user desires for a command (see the attached sample session). Although not readily obvious from the sample, commands may be performed in whatever order necessary to produce a desired result.

Not only is the "host language" presented in easily understood terms, data descriptions are also conveyed in non-technical language. Data sets are depicted as collections of attributes, or characteristics, pertaining to entities, or objects. Entities may be things ranging from employees or students to revenue and expense accounts to office space. Attributes may be any characteristic of entities--these attributes are mnemonically named to provide users an easy access mechanism to the pieces of information they require.

INTRODUCTION

Project FIND (Forecasting Institutional Needs at Dartmouth) has been established to provide a means for making institutional data more readily accessible to the college's officers and to develop models of the institution's operation to facilitate long-range planning. To satisfy these goals Project FIND has three objectives:

- (i) to provide a tool, requiring minimal computer expertise, allowing administrators and faculty to obtain current information about institutional data,
- (ii) to provide a universal format and language through which members of the college community can manage their private data in conjunction with institutional data, and
- (iii) to provide a planning tool for estimating the effects of changes in institutional policy. [1]

This paper focuses on the first two of these objectives--a computer system, also called FIND, for the management of data. This system is available to any member of the college community through terminals connected to the Dartmouth Time-Sharing System. As an institutionally recognized data management system, FIND has been operational for about a year. It is widely used by people in the offices of Personnel Administration, the President, Student Affairs, the Dean of the Faculty, the College Editor, the Budget Officer, Alumni Affairs, Investments, and Affirmative Action.

THE FIND SYSTEM

History

Project FIND was actually initiated in April, 1972, in a course taught by President Kemeny. The efforts of that course produced the first version of the information retrieval system in June, 1972. At that time the Director of Data Processing became responsible for the further development of the software and the definition and inclusion of institutional data into the system. During the next three months, the original programs were refined; new ones were added to perform simple statistical analyses; and data bases including the college's

faculty and administration were cast in FIND's data format.

The 1972-1973 academic year saw the use of these data bases, and others with personnel and financial orientations, by the upper levels of management on an experimental basis. Because the original system had only retrieval, subsetting, and display capabilities, one of the primary programming tasks during this period was the design and development of a data base maintenance system to assure the timeliness of information contained within the system. In the fall of 1973, President Kemeny announced that FIND was to be the official repository of the college's institutional data and that the initial thrust would be in the area of employee personnel records.

In the past year this initial basic personnel data has been expanded to meet the varied needs of many sectors of the college. Data descriptions have also been standardized to assure uniformity of meaning of common data elements used in various offices. New commands have been added to the system as users have indicated the need for increasing flexibility and expanded capabilities. In fact, an interesting side effect of the implementation of the project's third objective--forecasting--has allowed the user community to write procedures which they needed for a specific purpose. Some of these user programs have universal applicability, and this fact was recognized by the FIND staff, who modified the programs to become integral parts of the FIND system.

The FIND Language

The FIND language provides its users a flexible, easy-to-use capability to access, modify, reorganize, and display information in permanent data bases. The language has been designed around a number of universal utility functions which are applicable to a number of administrative offices and provide the user

community with varying degrees of sophistication in their manipulations of a working data base. FIND commands may be grouped into five broad classifications: informative, primary (or elementary), advanced, miscellaneous, and maintenance. Each of the five general areas will be discussed and reference will be made to the use of many of the commands in the sample session shown in the Appendix.

Informative commands are designed to prompt the memory of casual users and to provide a short introduction to the novice user. Probably the most informative of these commands is EXPLAIN, which provides descriptions (briefly stated) of each valid command and of many additional key words used in reference documents. For the user who knows the general scope of his investigation (that is, which data base contains the desired information) but is unsure of nomenclature, the ATTRIBUTES command provides a display of each abbreviation available in a particular data base. To determine the institutional meaning of any attribute, one of two commands may be issued. The DESCRIBE command gives a brief explanation of an attribute and its characteristics; the DETAIL command provides the complete institutional definition and a list of all the possible codings (for attributes whose values have been shortened to reduce the storage requirements of a data base).

The primary commands are of the most importance to a user, for they provide the means of data access and manipulation. The first command normally issued in a FIND session is a RETRIEVE, which copies data from a permanent data base (described below) to a working data base. All subsequent commands act upon the working data base, allowing a user great flexibility without the opportunity to alter the permanent, or institutional, data base. The other primary commands

are SELECT, SORT, RESTORE, PRINT, and STATISTICS. Each of these commands specifies a type of operation to be performed on the working data base. The SELECT command allows the user to subset the working data base by providing a logical expression specifying the inclusion criteria for the subset. A working data base, or a subset thereof, may be reordered in ascending or descending alphabetical order according to any single attribute or group of attributes. (The sorting algorithm used treats all values as character strings and sorts these character strings according to their ASCII values.) Because the SELECT command causes the "working data base subset" to shrink, and the SORT command destroys the original order (or non-order) of a working data base, the RESTORE command is very important for it returns a working data base to its original form subsequent to a RETRIEVE command. Simple lists of entities in the "working subset" may be requested by a PRINT command. The list of attributes may be specified, in which case they are printed as requested, or no list may be given, and all the attributes will be printed as they are encountered in the working data base. Finally, the STATISTICS command provides the standard univariate statistical measures of a numeric attribute (mean, median, and range).

Advanced commands give a user the power of more advanced statistical techniques, the capability to combine attributes arithmetically, or to format more nicely a printed display. XTAB provides the user with simple frequency tables for one or two attributes; FIT provides linear or exponential curve-fitting techniques (regression); and CORRELATION provides bivariate correlations of attributes. The DEFINE command allows the user to create new attributes as a function of existing attributes, and constants, using simple Basic-like statements to convey the exact operations he wishes performed.

Finally, the `FORMAT` command allows control of the format (spacing, layout, pagination, page dimensions) of a printed page.

Miscellaneous commands provide users with amenities not entirely necessary to the existence of an information system, but which make life more bearable for the user. For example, a session at the terminal might have to be interrupted for some reason before the desired analysis or report is completed. To allow a user to "pick up where he left off," the `SAVE` and `LOAD` commands are lifesavers. `SAVE` makes a copy of the working data base (and any subset) at a particular instant, and `LOAD` when issued brings that copy back into the `FIND` system as the current working data base. Additional commands of this type include `RENAME`, `OUTPUT`, `LABEL`, `SCRATCH`, `COUNT`, `REDUCE`, and `TIME`. Many of these commands are self-evident. `RENAME` allows the user to change the name of an attribute to something more meaningful to him; `OUTPUT` provides the capability of directing printed output to a file rather than the terminal; `LABEL` allows the insertion of textual material at the head or foot of tabular displays. The `SCRATCH` command erases some or all of the attributes from the working data base, allowing the user to access multiple data bases in the same session with `FIND`. `COUNT` gives a user the current number of attributes, entities, and entities selected in the working data base. `REDUCE` causes the working data base to shrink to the size of the selected subset; this is most useful in the preparation of reports and data files for specific offices of the College. Finally, the `TIME` command tells the amount of CPU time used since the inception of the session.

Maintenance commands insure the accuracy of the various data bases. They are the vehicle for adding new entities to a data base, deleting entities no

longer in a data base (terminated employees, discontinued fiscal accounts, etc.), and the alteration of values for existing entities. A special command, UPDATE, signals FIND that the user wishes to be placed in the maintenance system, where he may issue the special maintenance commands: IDENT, LOG, ADD, DROP, ALTER, and MODIFY. IDENT is the first command issued in the updating scheme, and it tells the system which attribute (or attributes) the user wishes to use as identification attributes for the balance of the session (or until another IDENT command is issued). Because the UPDATE command and its associated commands alter only the working data base, it is necessary to provide a means for conveying information about updates to the permanent data bases: this is done with a LOG command, which specifies the name of a file in which all alterations to the working data base are to be stored. The LOG command causes recording of subsequent alterations only, any changes in the working data base prior to the issuance of a LOG command are not recorded.

Actual data modification occurs only when the ADD, DROP, ALTER, or MODIFY commands are issued. The ADD command adds an entity to the working data base. Upon issuance of the ADD command, only those attributes identifying the entity are added to the working data base; spaces are left for the subsequent filling of additional attributes using the ALTER or MODIFY commands. The DROP command causes an entity to disappear from the working data base; in its place a vacuous entity appears if the data base is restored. The ALTER and MODIFY commands allow the user to change data in existing records or to insert additional attributes for entities recently added. An ALTER command specifies a particular value for a single attribute to be changed, while a MODIFY command allows a user to enter values for a list of attributes for a single entity. Because of this

difference in structure, the ALTER command is designed to change the particular value immediately; the MODIFY command "stores up" its lists of values to be altered until the user specifies that updating should occur. Once an UPDATE session is completed and a user wishes to return to the information retrieval and display system, an EXIT command is given.

A Sample Session

The session depicted in the appendix is a derived example designed to "show off" various system features; thus it does not necessarily depict the way the system is in fact used at Dartmouth College. However, the data are taken from our current personnel information systems and do reflect the current status of the college. Two distinct applications are portrayed in the sample: first, a statistical summary of minority and female composition of the college's support staff is done, showing the sex and race distributions in various job-related and wage-related classifications; second, a departmental display is presented, where seniority is defined, and the display is based on a person's seniority in the department.

If we dissect the sample, we find that the first three commands are information-gathering in nature. They show us the range of data bases from which we may choose, the attributes in a particular data base, and the definitions of certain selected attributes from that data base. The fourth command shows a RETRIEVE command, in which we are retrieving from the STAFF data base seven attributes. The next series of commands involves cross-tabulations and frequency counts enabling us to determine the composition and distribution of minorities and women in this particular segment of the college's labor force. An analysis of these data is left to the reader.

Beginning at the eleventh command, we are narrowing the scope of our analysis to a single department (the Data Processing Center staff), defining a person's length of service as a function of his date of hire and the current time, defining his tenure as a function of the date on which he was promoted to his current job, then sorting on tenure and displaying all information in the working data base about the individuals in the data-processing department.

DATA BASE STRUCTURE

A data base is a structured collection of information. For example, a data base may contain biographical data about individuals, financial data about the institution (or some segment of it), or information about student's grades in courses. Our data structure is a rectangular matrix (or two-dimensional array) structure. Each row in the structure represents one entity or observation of a data base. An entity may be a person, a financial account, or a course. Each column in the data structure is an attribute or characteristic of an entity. An individual's sex, race, or salary are examples of attributes. The value of an attribute for an entity is called an item. See Figure 1 for a schematic of a data base.

Figure 1
FIND Entities, Attributes and Items

		attributes			
		name	age	rank	address
entities					
				item	
			item		

Functionally, a data base is a collection^{ed} of files with an index, or directory file, locating each particular attribute in a specific data file. In addition to an attribute's name and file location, the directory also contains information about its protection level, its availability as a unique key to an entity, its length (maximum), and its type (whether numeric or alphanumeric). Because of this type of structure, it is very convenient to provide attributes with meaningful names, such as RACE (for an individual's race), HOURLRATE (for an hourly wage rate), or BUDNUM (for a fiscal account's budget number). We have applied an eight character limit to attribute names, thus necessitating some abbreviations. The abbreviation concept has also been carried over to attribute values, a fact alluded to in the discussion of the DETAIL command, to try to minimize the amount of space necessary to store data item values.

The data files themselves are inverted images of a data base. Logically each row of the data file is an attribute and each column is an entity. Thus statistical analyses of a series of attributes become easy retrieval tasks--the system is required to retrieve exactly the quantity of information needed to perform the task. Obviously, an employee profile requires retrieval of all information for all employees, an expensive task for a single profile.* These data files are direct-access files, stored on magnetic disk units, to minimize the search time necessary to locate an attribute and subsequently read its item values into core memory.

All data, irrespective of type, length, or protection level, are stored in the logical attribute blocks within the data files. Data of any of these

* This quirk of the system is currently undergoing a dramatic change. Effectively, the RETRIEVE command is being altered to allow a selection expression as part of its argument list, thus allowing a selective retrieve of the subsequent attributes based upon the selection criteria.

types may be stored in the same file, and the system will recognize its peculiarities and decode to a valid data item for the working data base. Thus sensitive protected data, which may be either string or numeric, will be decoded and unpacked if necessary to provide items for the user (see Figure 2).

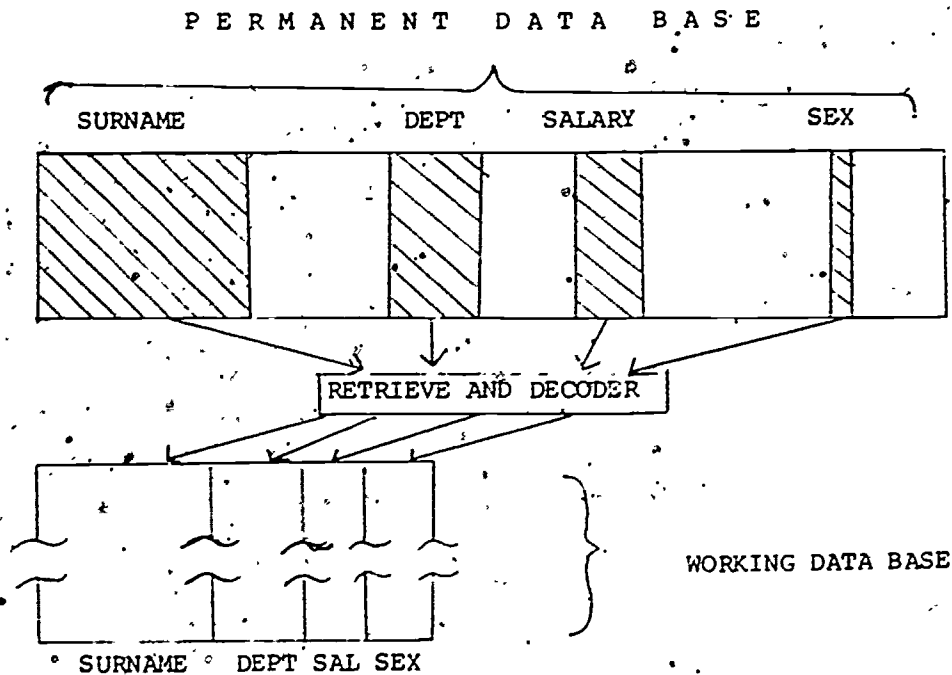


Figure 2
Relationship Between Permanent and Working Data Bases

Once retrieved, the working data base is again organized as the standard rectangular matrix. Additionally, the working base includes an order list which is created by the SELECT and SORT procedures and is then used as an indirect pointer to the working data base by all other modules (see Figure 3). This is done to alleviate the time-consuming problem of copying the complete working data base whenever a subsetting or reordering of the data base is requested by the user.

Figure 3
Working Data Base Structure

Pointer List	Working Data Base Entities	
	Surname	SEC
3	SMITH	MB
4	JONES	FC
2	BROWN	FC
1	DOE	MC

This pointer list orders the working data base alphabetically by surname.

CRUCIAL ISSUES

Before FIND could be regarded as an institutional tool for the management of information, a number of procedural issues had to be solved. One of the earliest issues was that of the definition of institutional data. In the 1973-74 academic year, a task force directed by the Institutional Research Office surveyed the college community to ascertain what data elements were currently being captured, the form they took (machine-readable or not), how they flowed through the various offices of the institution, and also asked the question of what other data elements were necessary for the general information needs of the college. From that task force, a series of working papers listing and defining the various elements of institutional data were prepared. The FIND staff, in conjunction with various offices around the campus, has made heavy use of the results for the capture of data in FIND-accessible formats.

A second issue to be resolved was the treatment of sensitive data. How should a person's salary be protected from the prying eyes of his counterparts? The scheme which evolved for protection of sensitive data places the complete burden for secrecy upon the individual offices with need to know protected information. An encoding-decoding scheme using a word as a seed to a random

number generator was developed whereby the seed word was never stored in the computer. If a user loses a data password, there is nothing for him to do but recreate the passworded information from its original sources!

The enciphering technique works in the following manner. Since FIND's data records are stored in an inverted structure on disk, all sensitive data of a particular type are stored in contiguous locations. Thus, a sensitive attribute, such as salary, is stored as a series of n-character values. If salaries are five digit values and there are 500 people in a data base, the string of salary information will be 2500 characters in length. Given the data packing technique employed, this information is stored in 278 contiguous 36-bit words in the FIND system.

The data password supplied by the user is an eight character (72-bit) quantity. This password becomes the first in a series of 278 pseudo-random numbers. From each of these pseudo-random numbers 36 bits (one word) are extracted and used to decode one word of the salary data. (See Figure 4 for a schematic description of this technique.) The decoding operation is an "exclusive or" applied to the encoded word. Because this operation is a reversible transformation, exactly the same procedure (and the same 72-bit password) is used to encode the data when a data base is created. [1]

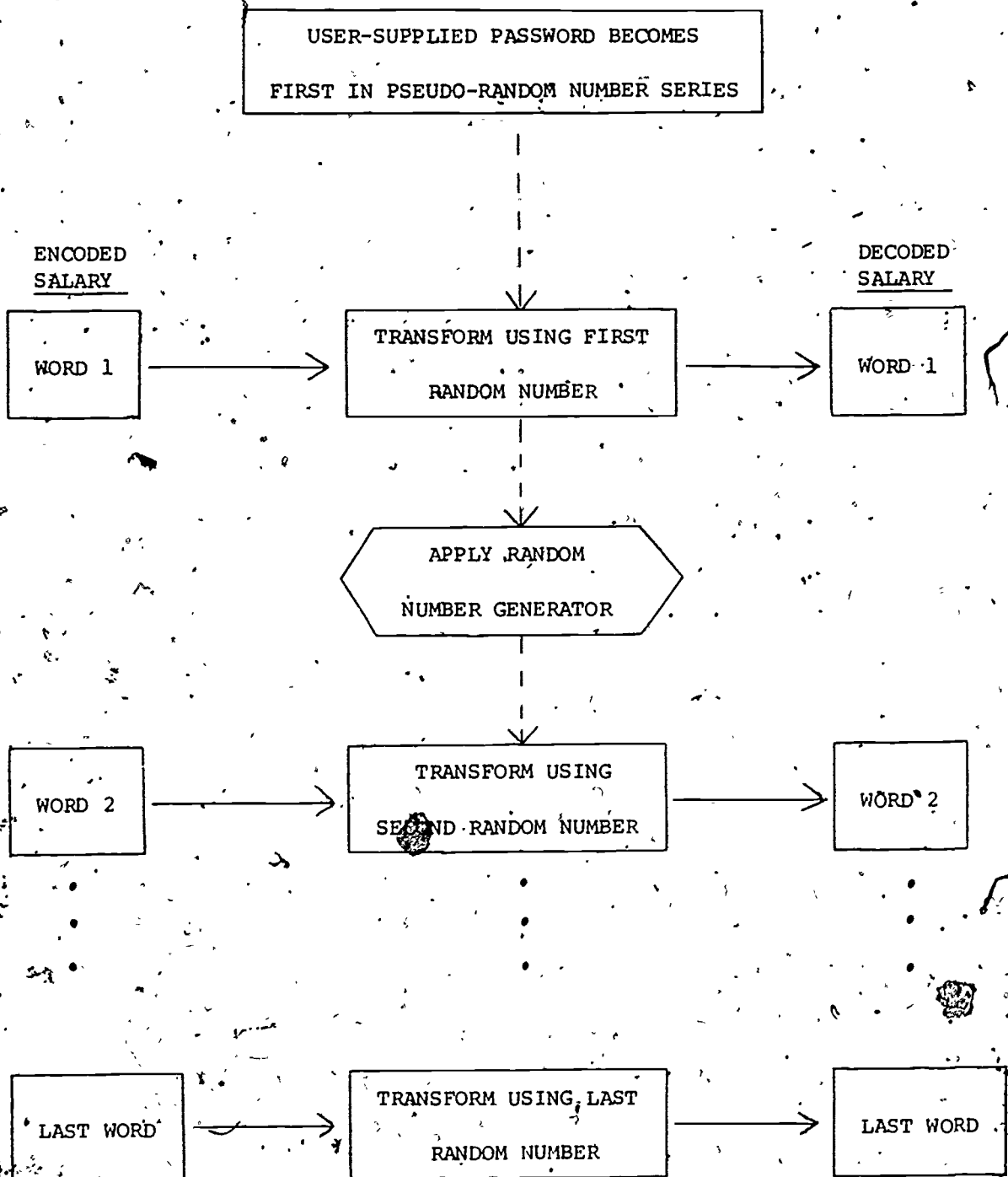


Figure 4
FIND Enciphering Technique

Appendix I A Sample Session.

OLD DPCLIB***:FIND
READY

RUN

FIND (COMPILED), 04 DEC 74 08:26

PRIVACY WARNING.

FIND HERE!

? EXPLAIN DATABASE

DATABASE
=====

Presently (2/11/74) there are twelve data bases publicly available to users of FIND:

ADMIN	Administrative Officers of Dartmouth College
ART	Catalog of College-owned Art Objects
BUDGET	Current Year Budgetary Information for the College
DART77	Biographical data on Dartmouth Students, Class of 1977
DEMO	FIND Demonstration Data Base--DTSS, Inc.
ENDOW	Endowments of the College and their Budgetary Distribution
FACULTY	Faculty Members of Dartmouth and its Associated Schools
OTHERS	Dartmouth's Temporary Employees
SERVICE	Dartmouth's Service Trades Employees
SLOAN	Educational Financing by Graduates of Selected Institutions
SPACE	Inventory and Utilization of College Building Space (200)
STAFF	Dartmouth's Non-unionized Staff Employees

DONE

? ATTRIBUTES STAFF

STAFF ATTRIBUTES ARE:

BRTHDAY	BRTHMNTN	BRTHYEAR	CITIZEN	EMPSTAT
EMPTYPE	FINDNUM *	HIREDATE	MAILADD2	MAILCITY
PARTFULL	RACE	RANKDATE	REGTEMP	SEX
STATMNTN	STATYEAR	UNFVCODE	COLCODE	FRSTINIT
MAILSTE	MAILZIP	FRSTNAME	FULNUM1P	FULNUM2P
GRADE	SALANOWP	EMPNUM *	SOCSECN	HRSWK
MIDINITS	SUFFIX	SURNAME	TERMDATE	TIAA
HOMECITY	HOMESTE	HOMEADD	FULLNUM1	FULLNUM2
EECODE	REPORTS2	HINBOX	MONTHS	DEPT
FULNUM3P	FULLNUM3	SUBDEPT	DIV	SPOUSNAM
WORKADD	WOCITYST	HOMEPHON	JOBTITLE	VISA
MAILADD1	COLEDSP	WORKPHON	DEGRHIGH	HOURLATE+
QRATNOW +	SALANOW +	EVAL74 +	EVAL73 +	HOMEZIP

INDICATES PASSWORD PROTECTION

INDICATES UNIQUENESS

DONE

? DETAIL EEOCODE, GRADE

EEOCODE

This is a one-digit numeric code to indicate the individual's determination in accordance to the standard classifications under "Equal Employment Opportunity" classifications, as follows:

CODE	DESCRIPTIONS
1	Officials and Managers
2	Professionals
3	Technicians
4	Sales Workers
5	Office & Clerical
6	Craftsmen (Skilled)
7	Operatives (Semi-Skilled)
8	Laborers (Unskilled)
9	Service Workers

GRADE

A two-digit numeric code indicating the individual's present grade, as per the Staff Personnel Classification System.

DONE

? RETRIEVE STAFF, GRADE, LEOCODE, RACE, SEX, DEPT, HIREDATE, SURNAME

*RET: 7 ATTRIBUTES RETRIEVED FOR 914 ENTITIES, LAST MODIFIED 11/11/74

DONE

? XTAB LEOCODE, STD

EEOCODE	FREQ
---------	------

0	6
2	119
3	138
5	594
6	12
7	6
9	37

914 IN SAMPLE, 2 EXCLUDED.

DONE

? XTAB SEX,STD,EEOCODE,STD

EEOCODE	3	5	9	2	7	6	0
SEX							
F	72	561	24	70	5	0	1
M	66	33	13	49	1	12	5

914 IN SAMPLE. 2 EXCLUDED.

DONE

? SELECT EEOCODE = 5.

*SEL: 594 ENTITIES SELECTED

DONE

? XTAB GRADE,STD

GRADE	FREQ
0	2
1	20
2	56
3	104
4	69
5	165
6	31
7	99
8	11
9	35
11	1
12	1

594 IN SAMPLE. 0 EXCLUDED.

DONE

? XTAB RACE,STD,SEX,STD

SEX	F	M
RACE		
C	552	30
H	0	1
B	5	1
S	1	0
A	1	0

594 IN SAMPLE. 3 EXCLUDED.

DONE

Vol. II

? RESTORE

*RES: 914 ENTITIES RESTORED

DONE

? DEFINE SENIOR = 741231 - HIREDATE

DONE

? SELECT DEPT = "DPC"

*SEL: 12 ENTITIES SELECTED

DONE

? SORT -SENIOR

DONE

? PRINT SURNAME, HIREDATE, SENIOR, GRADE, EEOCODE, RACE, SEX

SURNAME	HIREDATE	SENIOR	GRADE	EEOCODE	RACE	SEX
MARTIN	601101	140130	9	3	C	M
HANNAH	620103	121128	6	3	C	F
JORDAN	651025	90206	12	3	C	M
GUNN	660912	80319	9	3	C	M
CHAPIN	690701	50530	13	3	C	M
FIFIELD	700113	41118	9	3	C	M
MARLER	700401	40830	9	3	C	F
GOODHUE	700720	40511	6	3	C	F
STILES	710803	30428	5	5	C	F
HOOSE	730110	11121	12	3	C	M
BENHAM	730528	10703	6	3	C	F
LESKER	740909	322	12	3	C	F

DONE

? STOP

80.463 SEC. 1858 I/O

Appendix II FIND Commands

Informative Commands

ATTRIBUTES	lists attributes in a requested data base
DESCRIBE	provides brief explanation of an attribute
DETAIL	provides complete explanation of an attribute
EXPLAIN	provides brief explanation of commands and keywords

Primary Commands

PRINT	displays a list of attributes
RESTORE	restores a working data base to its original non-selected, non-sorted state
RETRIEVE	accesses a permanent data base and extracts attributes for a working data base
SELECT	selects a subset of the working data base, according to specified criteria
SORT	sorts the working data base, according to specific attribute(s).
STATISTICS	provides summary statistics for an attribute
STOP	terminates a session with FIND

Advanced Commands

CORRELATION	provides correlations between two attributes.
DEFINE	allows creation of a new attribute by combination of existing attributes and constants
FIT	allows linear or exponential regression analysis
FORMAT	provides flexibility in determining output format
XTAB	allows one- or two-attribute tabulations

Vol. II

Miscellaneous Commands

COUNT provides a summary of attributes and entities in the working data base

DEBUG provides a debugging aid for FIND personnel

EXECUTE specifies a file from which FIND commands will be read and executed

LABEL inserts text into output

LOAD reads specially-formatted files to recreate a working data base

OUTPUT specifies a file to which output will be directed

REDUCE shrinks working data base to entities currently selected

RENAME changes name of an attribute in working data base

ROUND rounds numeric attribute to specified places (forcing alignment of decimal point)

SAVE stores a copy of current working data base in specially-formatted files

SCRATCH erases some or all attributes from working data base

Maintenance Commands

UPDATE initiates a maintenance session

ADD adds an entity to working data base

ALTER changes the value of an item in working data base

DEBUG provides a debugging aid for FIND personnel

DISPLAY prints selected attribute values for an entity

DROP deletes an entity from working data base

ECHO	"parrots back" the most recent changes performed by ALTER or MODIFY .
ESTABLISH	creates filler in working data base for specified attributes
EXIT	terminates a maintenance session
EXPLAIN	explains commands and key words
IDENT	specifies list of attributes to use for identification
LOG	specifies name of file to record changes for subsequent application to permanent data base
MODIFY	changes items for specified entities
SCRATCH	erases contents of specified file
STOP	terminates a session with FIND

References

- [1] John S. McGeachie and Donald L. Kreider, Project FIND--An integrated information and modeling system for management, ACIPS--Conference Proceedings, Vol. 43, 1974, 529-535.

A MODEL FOR COMPUTER CENTER BUDGETING

John J. Falcone
Director
Computer Center
University of Delaware

Anthony Graziano
Assistant Provost
Budget & Analysis
University of Delaware

Joseph Remy
Associate Director
Computer Center
University of Delaware

A MODEL FOR UNIVERSITY COMPUTER CENTER BUDGETING

John J. Falcone, Director

Joseph A. Remy, Associate Director

Anthony F. Graziano, Assistant Provost for Budget and Analysis

University of Delaware, Computing Center

Smith Hall, Newark, Delaware 19711

Allocation of computing funds in universities is a difficult task, especially in times of tight government and state budgets. At the University of Delaware we have come through a two-year period of high development of administrative systems. This development has produced many new systems. This paper presents a simple matrix which is used for budgeting the machine time, people, and equipment necessary to maintain and enhance these systems as well as to develop new systems in an organized fashion.

1. INTRODUCTION

The internal allocation of funds for computing is a difficult task and oftentimes is confusing to the user of computing services. Budgeting and allocating the computing resource is often treated as a single-dimensioned entity while, in fact, it is multi-dimensioned. One dollar of computer funds allocated to a user can obtain production time for an established system; maintenance (programming and machine time) to a system; or the programmer and machine time to develop a new system. That same dollar could be spent on terminal hardware intended to enhance the operating, maintaining or developing of a system. Figure 1 is a graphical representation of the activities or parameters that might better define the allocation of the computing resource.

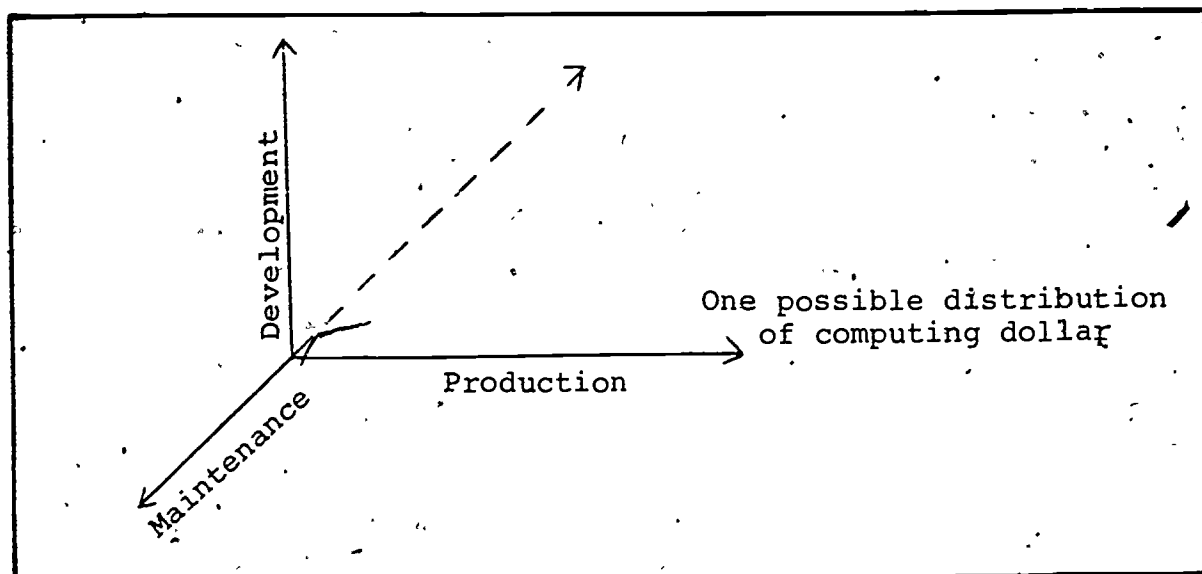


Figure 1. Computing Dollars by Type of Activity

Previous allocations of the University of Delaware computing resource and the proposed '74-'75 allocation of that resource can be represented graphically using the above axis to depict a two-year evolution of the allocation from a high development investment to a high production investment (Figures 2 and 3): Unless we are prepared to sustain exceptionally large and continuing increases in the University's budget for computing, resources devoted to investment in the design and construction of functioning applications must ultimately be devoted to sustaining the intent of the investment. And unless some continued steady-state allocation of development and maintenance is provided to enable the computing resource to renew and enhance its services over time, the whole cycle of high cost development will have to be repeated periodically in a most inefficient way. The development services of the Computing Center should be planned as a steady flow of assistance targeted for those areas where the University as a whole will benefit most. It cannot be a crash

program required every five years or so to plug large and immediate demands arising from unanticipated needs. The steady flow of new development should result from Computing Center leadership guided by user dialogue and understanding. The users must continually participate in dialogue amongst themselves and with the Center; they must understand that the University and its budget for computing is not a "funny money" key to unlimited and unconstrained demand; and they must participate in the determination of what is best for the University as a whole even at the expense of their individual needs and aspirations.

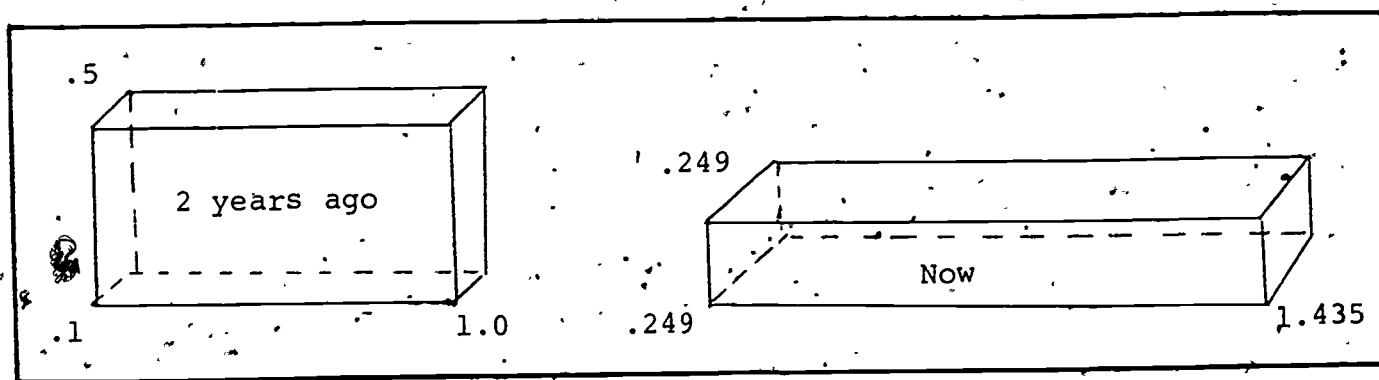


Figure 1. Dollars in Millions

Figure 2.

2. DEFINITIONS AND CONCEPT

Table A defines the terms used in this discussion.

Figure 4 is a 3 x 3 matrix which represents the three activities involved in delivering computing services versus the resources required to accomplish them.

Activity Resource	Production	Maintenance	Development
People			
Machine			
Equipment			

Figure 4. Activity vs. Resource Matrix

A computer dollar representing some measure of computing service demand can be deposited in one of nine cells; i.e., we could invest the dollar in people for new development (Row 1, Column 3) or we could invest the dollar in machine time for production (Row 2, Column 1) or any of 7 other possible ways. However, a wise investment in computing dollars considers the trade-offs implicit in selecting any one of these activities over the others as well as the resource-cost to achieve them. Insofar as the University is concerned, the computing resource is limited by the combination of people, machine time, and equipment which can be purchased with the total university allocation for computing. We, as administrators, need to introduce users to these facts of life by allocating to each of them some equitable share of and control over a piece of the total resource. And we need to monitor their decisions as well as provide to them the best and most informed leadership available through central coordination at the Computing Center.

3. THE PROCESS OF COMPUTER SYSTEM DEVELOPMENT

Let us consider the typical process of new system development. A specific need is identified by a department or perhaps some university-wide problem is perceived which conceptually would lend itself to computerization. The development of such a system is usually considered in the form of a feasibility study. This study considers the initial cost of system design, programming, and documentation as well as the on-going production (operating costs). Enhancements or maintenance of the future system are not usually considered for it would be difficult, if not impossible, to anticipate this activity. Therefore, we have unavoidably built in the first possibility for a future budget overrun when the system requires later change or modification in its production mode.

If the feasibility study is accepted, the development work begins. At this time, a group or project team is designated, a schedule of machine time is developed, and any required special equipment is ordered.

As the system is developed, the actual machine requirement for future production becomes evident. When the system reaches operational status, the full attention of the project team is no longer required and the people resource required to operate the system in a production mode is considered in the machine cost "overhead". Production support, correction of errors, standard keypunching, etc., are also considered in this machine cost for production.

When the system reaches the production mode, a certain tolerable level of enhancements or changes are expected. But these can be serious and disruptive if the system has not been designed or implemented to meet clearly stated objectives of the user. If organizational policy is constantly in a state of flux, a particular system may experience significant requirement for enhancement or modification regardless of how well the particular user may have originally specified design parameters. System flexibility that could have been anticipated in the planning and design stage, but which is not provided for in the delivered system, can well result in high maintenance activity, user and community disaffection, and high costs of people and machine resources attendant to modification of the system.

Finally, when a multitude of changes have so corrupted the original design, or when the system no longer meets current policy or needs, a demand is created for its replacement. At that time, we return to a feasibility study for a new computer system including considerations of the costs as well as new relationships with other existing systems that have been developed over the life span of the system in question.

It is not easy to determine when a computer system has outlived its usefulness. Such a consideration must take into account any and all systems in existence at that time or being considered for implementation at that time. The proposed concept of a budget matrix

forces consideration of all interrelated costs and relates these costs directly to the current inventory of services and the planned allocation of the total computing resource.

In addition to the project development and operating costs of a proposed system, maintenance costs must be considered or provided for in a manner which motivates stability and sound judgment. Maintenance costs are less deterministic than development and operational costs, and to a large extent, are a function of the user's total computing resource available. It should also be possible for a knowledgeable investor to allocate one dollar toward maintenance in order to secure a more efficient system, to reduce his operating costs in an area by, say, 10 dollars, thereby leaving 9 dollars worth of computing resource available to everyone for more development and/or more production work. Overall cost effective utilization of the University's total computing resource can only be achieved if each user or subset of users understands the budgeted limitations of the total resource and participates in the allocation of that resource to the functions of production, maintenance and development.

4. USING THE ACTIVITY/RESOURCE MATRIX FOR PLANNING

Certain areas of the Computing Center may never be directly allocated in this manner; e.g., Academic Services, Systems Software. Indeed, it may always be necessary and appropriate to recover the systems programming manpower costs as production "overhead". The demands of the academic community at this university are still in a state of change and several new and large requirements appear to be

evolving. Until the Center is better apprised of the current and future needs in this area, it would not be prudent to allocate development and/or consultation costs. We do not know enough about the needs of the user community and assigning these efforts in a rigid way at this time may overlook targets of opportunity that bring benefits to the total computing community or a large segment of that community.

Eventually, however, the University community benefits from a planned allocation and utilization of the non-production elements of the Computing Center total resource. New development projects and major maintenance projects can span the significant portion of a year. The size of our Computing Center staff available in this area prohibits the wasting of any portion of that resource that can be tied up in meetings and other discourse arising from the confusion naturally attendant to meeting crisis requirements. The state of the economy also dictates efficiency. For the sake of the total university community, developmental and maintenance projects must be planned with an eye toward encouraging growth and efficiency in computing by allocating development dollars into the computing budgets of academic and administrative units who are now comparatively computer-destitute. Development dollars must be expended toward increasing the quality and quantity of productive systems in order to promote a positive growth rate of the University's computing community. And until that growth rate is known and intelligently anticipated, it is difficult to plan knowledgeably about future requirements.

It is proposed that the boundaries of administrative computing be fixed for the coming year, these boundaries being somewhat flexibly assigned to each Vice-President responsible for some specific set of services. In addition to allocations for production costs anticipated for each of his operating units, each Vice-President is assigned a recommended total budget for maintenance with which he can:

(a) increase the efficiency of existing systems in order to reduce future costs of their production outputs, or (b) attack short-term problems arising from new external or internal demands for information and controls. In short, each Vice-President has his own "mini" computing center complete with people, computer (pro-rated) and equipment.

Additionally, the Provost, as Chief Budget Officer, has been assigned the development portion of the total computing resource to promote block-by-block construction of the total campus planning and management information system. These funds are to be targeted for projects such as space utilization, major enhancements in financial and student data systems, and other, perhaps nonexistent, operating systems necessary to the management of the University's functions.

Graph I represents the allocation schema proposed at the University of Delaware.

The plan, to date, has met with favorable comments. It is of interest that each participant has developed an acute awareness of the nature of the computing resource. We hope to build upon this awareness an involved and economically conscious user community..

TABLE A

1. Resources - that which is necessary to develop, operate and maintain a computer system..

1.1 People - programmers/analysts/systems personnel who will design, operate and change computer systems.

1.2 Machine - the amount of computer time and pro-rated operations personnel necessary in the running of existing systems, effecting changes or enhancements to existing systems, and the development of new systems.

1.3 Equipment - all peripheral (terminal) equipment necessary primarily to run on existing system but also needed for design and changes to proposed or existing systems.

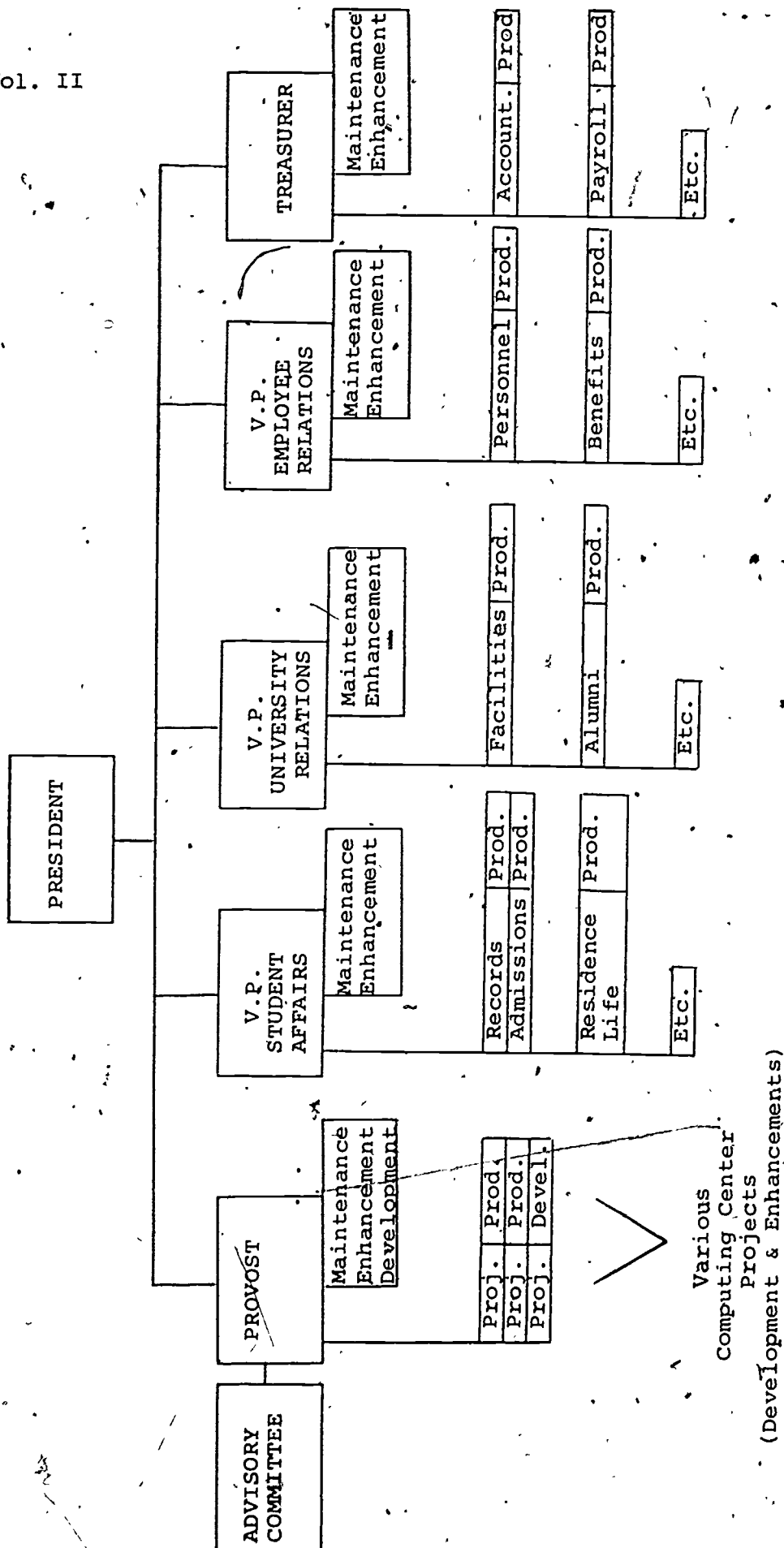
2. Activities - the three components necessary to create, use and change a computer-based system.

2.1 Production - that activity which deals with the successful operation of a previously developed system.

2.2 Maintenance - that activity which deals with short-term changes or enhancements to an existing system.

2.3 Development - that activity which causes a need for a computer-based or computer assisted system to come into being; i.e., design, programming, documentation, etc.

MAINTENANCE & DEVELOPMENT DISTRIBUTION CHART



GRAPH I

ADMINISTRATIVE INFORMATION THROUGH
ON-LINE SYSTEMS

Dovalee D. McElroy
Systems Analyst
Office of Institutional Research
The University of Texas at Dallas

Eugene E. Payne
Director
Planning and Management Systems
The University of Texas at Dallas

ADMINISTRATIVE INFORMATION THROUGH ON-LINE APL SYSTEMS

D. D. McElroy
Office of Institutional Research
The University of Texas at Dallas
Richardson, Texas 75080

E. E. Payne
Director of Planning and Management Systems
The University of Texas at Dallas
Richardson, Texas 75080

The University of Texas at Dallas is a developing institution. In preparing for a new mission, that of starting a large undergraduate program, the administration has found it necessary to solve many new planning problems while, at the same time, continuing to operate a complex program of sponsored research and graduate studies. The Office of Institutional Research has assisted the executive administration by (1) conducting special quantitative studies that require quick answers; and, (2) developing information systems that make management information available on short notice.

This paper briefly discusses the mode of operation of the Office of Institutional Research and describes several studies in which on-line information systems were developed. The information systems described include: Facilities Management, Evaluation of Service Organizations, UTD Economic Planning Model, Classroom Utilization, Analysis of Community College Data, Report Monitoring, A Text Generation System, and Budget Analysis.

1. INTRODUCTION

The University of Texas at Dallas is an institution in the final stages of a complete metamorphosis. It was established, and has been internationally recognized for over a decade, as a leading research foundation with teaching activities only in graduate studies. In 1969, the scope of the institution was significantly expanded to include degree program studies for undergraduate students. This year, the University is completing the final

stages of a large building and staffing program, and is preparing to open for 4,000 undergraduates in the fall of 1975.

In recognition of the significant planning and development problems that faced the institution in preparing for its new mission (while at the same time continuing to operate a complex program of sponsored research and graduate studies), three years ago the administration at the University organized the Division of Planning and Management Systems (see Figure 1). This organization included a function new to the University, the Office of Institutional Research. Institutional Research was charged with the responsibility of providing special quantitative staff support to the executive administration (the President and Vice Presidents). The Director of Planning and Management Systems (who also served as the Director of Institutional Research) established the following operating guidelines for the new organization:

- Institutional Research will function along the lines of a small corporate operations research group. It will serve the Offices of the President and the Vice Presidents.
- Emphasis will be placed upon solving managerial problems. Studies which have limited value (except for publication) will be discouraged.
- Large, long-term projects will be limited to no more than 50% of the man-effort. Experience has shown that solving numerous small to medium size problems will assure a broad base of service and a continuous high return for the effort expended.
- Staffing should be limited so that there is always more "critical" work than available manpower. This produces a natural selection

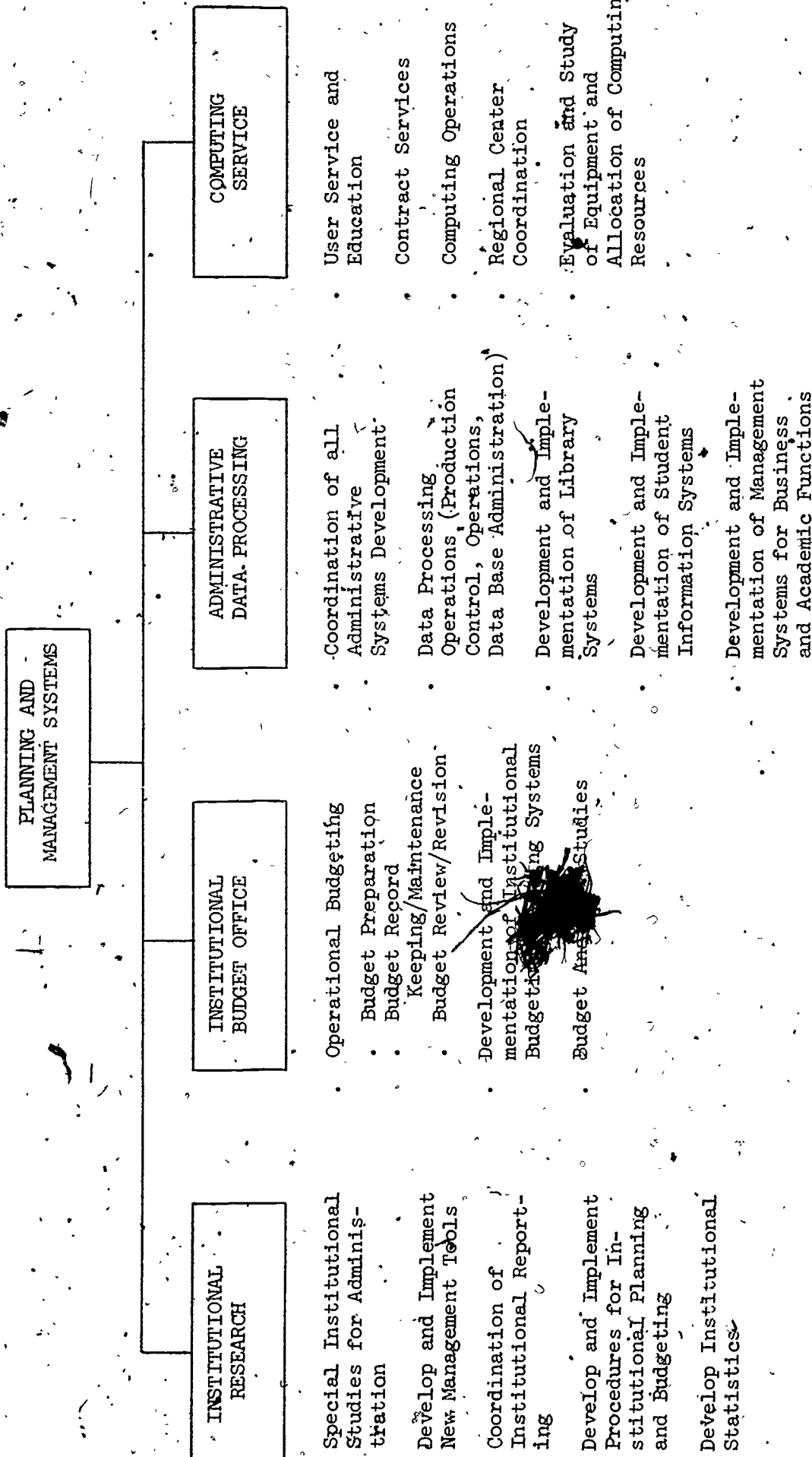


Figure 1. Functions of Planning and Management Systems

process (for staff organizations) which helps to assure that unimportant projects are seldom initiated. The staff will consist of a small, permanent core of professionals which will be supplemented by graduate students and faculty on temporary assignments.

From the point-of-view of the executive administration at the University, this mode of operation has proven to be very effective. Numerous management problems have been effectively dealt with because managers were furnished the necessary information on a timely basis.* This timely information was frequently made possible because of the ability to quickly develop and use computerized on-line information systems. This paper explores examples of Administrative Information Systems that have been developed and used at UTD. The systems that are discussed are:

- Facilities Management
- Evaluation of Service Organizations
- UTD Economic Planning Model
- Classroom Utilization
- Analysis of Community College Data
- Report Monitoring
- A Text Generation System
- Budget Analysis

Several of these applications are rather common place, but are of interest in providing a spectrum of the information systems that have been developed and found to be of high utility.

* "Better information does not lead to better decisions, unless the managers are highly capable and they have the opportunity to take advantage of the "better" information."

2. EQUIPMENT AND FACILITIES

The small staff of the Office of Institutional Research works closely with the executive administration. In effect, the Institutional Research staff has frequently become the conduit which provides the administration access to quantitative management information. The basic computing facility used by Institutional Research to carry out its activities is an IBM 370/155, which is utilized through either an interactive terminal, or a Remote Job Entry (batch) terminal.

Although several different computer languages are used, the greatest success in meeting the demand of quickly developing management information systems has been with APL.* This language has several advantages:

- Easy to use
 - Can quickly develop systems and bring into full operation
 - Has powerful matrix operators which are useful in modeling
- All of the applications discussed below were developed using APL.

3. FACILITIES MANAGEMENT

The planning, control, and allocation of space was a significant management problem that was growing worse. An on-line "Facilities Management" system was developed and implemented to assist administrative decision makers. Some of the features of the system are:

- A complete listing of the attributes (size, equipment, assignment, etc.) of any one room, or of all rooms, is always available from a terminal.

* APL is a language originally developed by Kenneth E. Iverson and described in his book, A Programming Language. [4]

- . A "formula" model has been developed to assist in the annual budgeting of space. Budgetary units are allocated space on a "formula basis" (standard). Formulas are based upon personnel, semester credit hours, research, and equipment.
- . A special request from a budgetary unit, resulting from an unforeseen event, for additional space can be instantly compared against both "formula" (standard) and the institutional average for similar budgetary units.
- . The system provides the ability to "game" or to project future space needs based upon enrollment and staffing projections.

The flowchart in Figure 2 illustrates the logic of the Facilities Management System. A more detailed description of this system is provided in the Proceedings of the 1973 CAUSE National Conference, [1]

4. EVALUATION OF SUPPORT DEPARTMENTS

Since 1972, at the request of the Vice President for Business Affairs, an evaluation of support organizations has been conducted every six months. The first evaluation covered 22 support departments such as mail, supply, computer services, cafeteria, bookstore, print shop, etc. The evaluation proved to be a useful source of information on both areas of problems and exceptional performance. Since it was started, various administrators, perhaps suspecting a problem area in their organization, have asked to be included in the confidential survey. The last evaluation included 31 separate departments.

4.1 Description of the System. The evaluation is based upon a survey of the opinions of the administrative heads of the various departments that are "users" of the support departments. A simple questionnaire allows the user

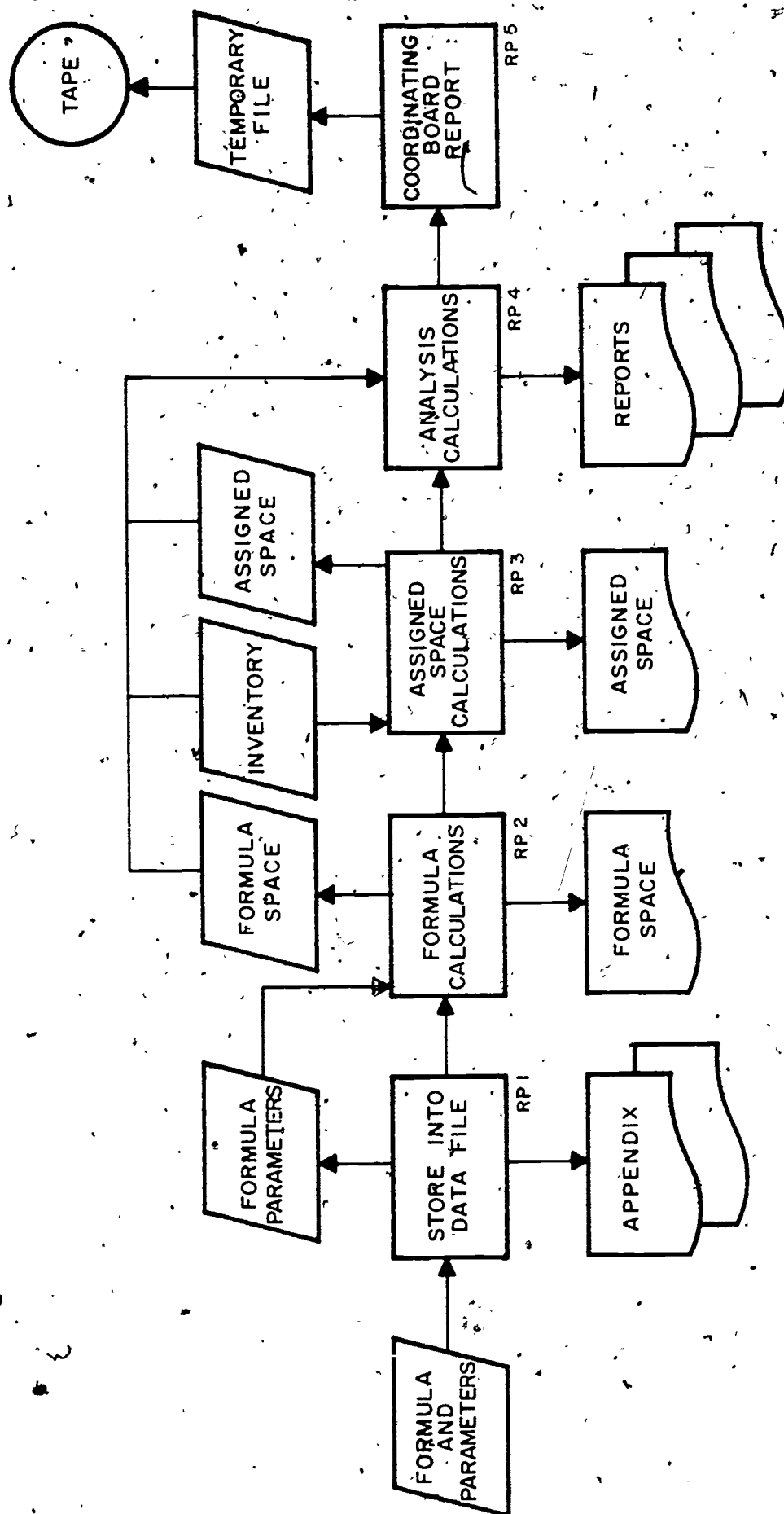


Figure 1. Facilities Management System

to evaluate the support departments on a scale of 1 to 7 on (1) effectiveness, (2) efficiency, (3) cooperation and tact, and, (4) extra effort. There are two more questions which are used to indicate the level of interaction and dependence of the user on the support department. Comments are also encouraged.

An evaluation package is sent to the administrative head of each organization at UTD. The administrative head is encouraged to perform the evaluation by way of a brief panel discussion with the members of his organization that actually have contact with the support departments.

4.2 Reports. The questionnaires are keypunched and read into an APL file. The APL program then produces five reports on each support department.

1. Analysis of Data. This report gives frequency response count, mean, standard deviation, and tests for statistically significant deviation from average evaluation. It also reports the scores from the previous evaluation and the percent change.
2. Analysis of Data of Those Organizations Highly Dependent on the Support Departments. This is the same report as above only using those organizations highly dependent on the support departments.
3. Index of Evaluation. This report calculates two index numbers for each question and each support department. The index for a department is equal to the average score for the support department divided by the overall average for all support departments. Therefore, an index of 1 is an average rating, an index higher than 1 is an above average rating, and an index lower than 1 is a below average rating. A summary report ranks all support departments against each other.

4. Comparison of Current Index Ratings to Previous Index Rating. This report compares each support department's overall index with the ranking of the previous six months and gives the percentage change.
5. Comments. A summary of the user comments is printed out for each department.

5. UTD ECONOMIC PLANNING MODEL

At The University of Texas at Dallas several models have been used to assist the administration in economic planning. UTD has been an especially fertile area for the use of this type of quantitative management tool since the institution will grow in size from 300 students in 1973, to 700 in 1974, to 5,000 in 1975 (most of these are already admitted), to 12,000 by 1980. This student growth means tremendous growth of staff and facilities, which, in turn, means that the administration has the opportunity to determine (by their decisions) what the institution will look like by the end of this decade. In this environment, quantitative planning models have already proven invaluable in assisting executive administration in (1) determining requirements (facilities, funds, faculty, staff); (2) investigating alternatives; and, (3) determining, and planning within, fiscal and facility constraints imposed by the State Legislature.

5.1 NCHEMS Models. The Office of Institutional Research experimented with the planning models developed by NCHEMS.* Although these provided some useful ideas, which were added to the basic UTD planning models,** the NCHEMS models.

* See References [2, 3].

** The definitions in the UTD Economic Planning Model have been changed to be consistent with the terminology used in the NCHEMS products which have become virtual standards.

were found not to be usable at UTD because: (1) UTD planners were already accustomed (spoiled, perhaps) to the UTD interactive models which provided quick response; (2) the NCHEMS batch models cost several times more to run; (3) the UTD models (written in APL) could be easily altered to represent new questions; and, (4) the UTD models considered the Texas "Formula Funding System" and generated facilities requirements. These observations do not reflect badly upon the NCHEMS products, but rather illustrate what NCHEMS itself says -- that their models are a teaching tool and a point of departure for serious users of quantitative planning models.

5.2 Structure of UTD Economic Planning Model. The UTD Economic Planning Model is an approximation of the financial relationships of students, faculty, staff, facilities support, and money. Some of the important elements are:

1. The objective of the model is to assist in the development of a viable financial plan by answering various types of "what if" questions.
2. Parameters are "constants" which are given or set by the decision-maker. Parameters represent the control that the planner (or the environment) has over the situation under investigation. They represent policy decisions, courses of action, or the environment. Examples of typical parameters in the models are:
 - FTE student enrollment (by degree program, by student level)
 - Faculty workload (by discipline, by level)
 - Faculty salaries (by rank, by discipline)
 - Faculty rank distribution (by discipline)
 - Class section size (by discipline, by level)

• Texas State Formula Funding levels

• ICLM (Induced Course Load Matrix)*

Parameters may change, and even become "output variables,"

a. depending upon what is being investigated.

3. Output variables are values that a decision-maker wants to know (results) and which may vary because of different policy decisions or courses of action. Although these may change (depending upon what is being investigated), typical output variables are:

• Semester credit hours produced (by level, by discipline)

• FTE faculty (by level, by rank, by discipline)

• Faculty productivity (SCH/FTE faculty, by discipline)

• Student/Faculty ratio (by discipline, by level)

• Faculty salary requirements (by rank, by discipline)

4. The logic of the model is a mathematical description of the relationships between, and among, the parameters and output variables. The logic is briefly indicated in Figure 3.

6. CLASSROOM UTILIZATION INFORMATION SYSTEM

Efficient use of classroom space has been a major concern -- there always seems to be more classes than classrooms. This simple information system provided a means of both inventoring classroom utilization and comparing the utilization against a recommended standard for each type and size of classroom. In this system the following definitions are used:

a. Weekly Room Hours (WRH). Weekly Room Hours is the number of hours during one week for which a room is available.

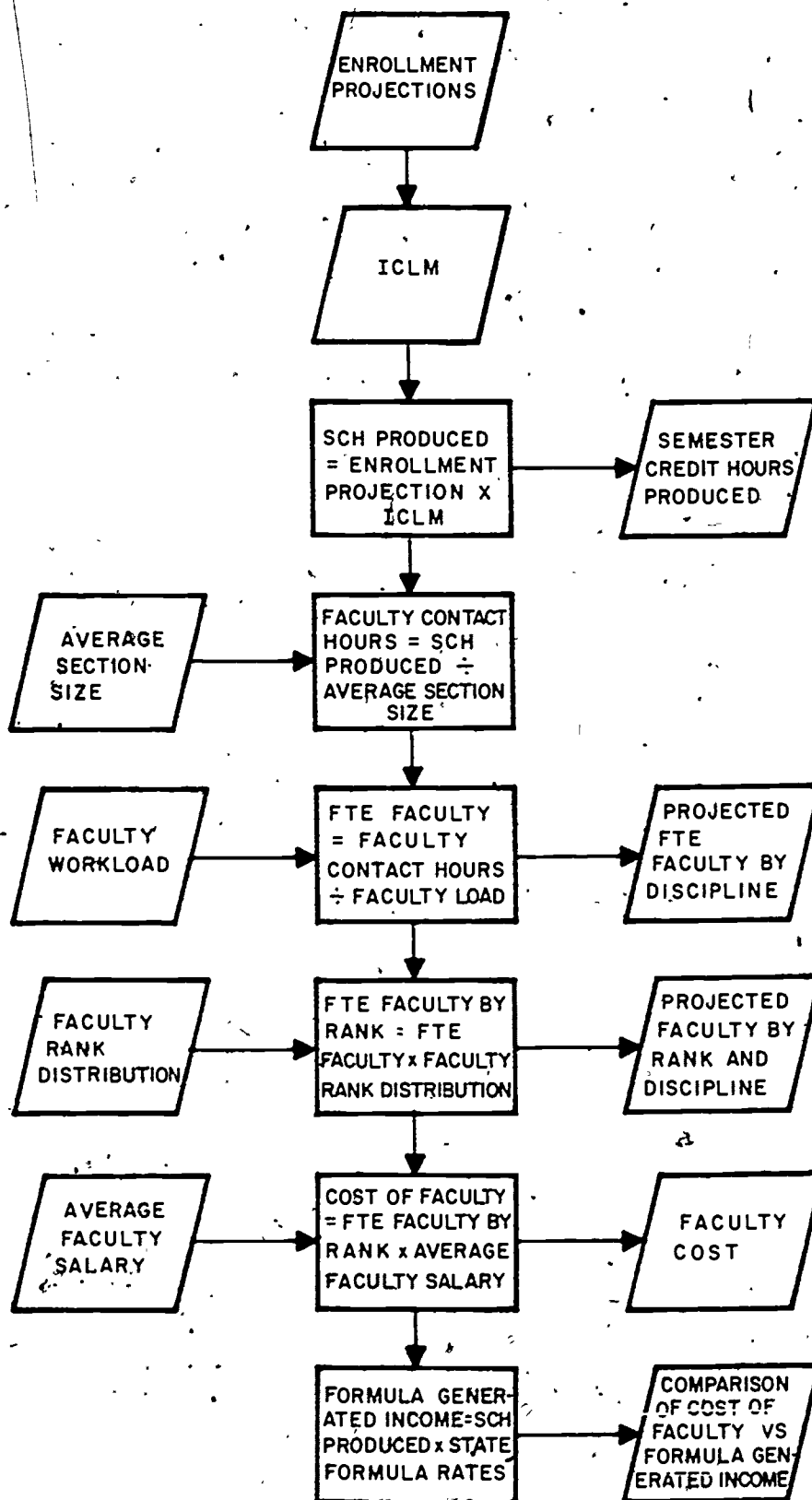


Figure 3. Basic flow of the TD Economic Planning Model

- b. Weekly Student Contact Hour (WSCH). Use of the room during one week by one student for one hour. (If a 3-hour class section has 30 students, then that section uses the room 90 WSCH.)
- c. Station. Station is a place for one student.
- d. Capacity. Number of student stations in one room.
- e. Station Occupancy. Station Occupancy = percent (variable) x number of WSCH.
- f. Utilization. Utilization = $WSCHs / (\text{percent} \times \text{capacity} \times WRH)$.

In this information system, the parameters are (1) the Weekly Room Hours (WRH); and, (2) the percentage utilization of station occupancy. Planners may establish utilization criteria (several sources recommend standards [6]), and then compare the actual utilization to these criteria (standards). For example, one may set a criteria of $WRH = 30$ and $\text{Station Occupancy} = .55 \times WSCHs$. Then, if a classroom for a particular semester has a capacity of 28 and $WSCHs = 378$, the utilization would be 82% ($378 / (.55 \times 28 \times 30)$).

The Room Utilization System provides a room by room analysis and a total utilization analysis. The room by room analysis has proven particularly useful in identifying rooms of both high and low utilization for exception management (action).

7. ANALYSIS OF LOCAL COMMUNITY COLLEGE DATA

UTD is an upper-level institution and will enroll many students from local community colleges. The Dallas County Community College System generates extensive student data for all of its four campuses each semester. These data have been made available to UTD. Therefore, an information system was developed to answer questions about the community college students and to

determine typical profiles of these individuals. The profile summary includes information on each campus, such as: headcount, total FTE students, FTE day and night students, percentage of students enrolled in transfer and technical programs, percentage of students which meet UTD entrance requirements and a breakdown of transfer students by major.

This analysis is performed each semester.. Also, it is planned to develop a student data base to analyze trends and compare to the profile of the entering students.

So far the data have indicated that the entering student will differ significantly from the "typical" college student. This information has proven valuable in planning academic programs, student life programs, and student services.

In addition to using the community college data to anticipate the type of student to expect, it is planned for this system to be expanded into a continuing analysis of the students that enter UTD. This will provide both UTD and the community colleges longitudinal studies of students, which will assist the planning and development efforts of administrators of both institutions. The cooperation of local community colleges has played an important part in implementing this information system.

8. A REPORT MONITORING SYSTEM

One of the responsibilities of the Office of Institutional Research at the University of Texas at Dallas is the coordination of all external reports for the University (i.e., HEGIS, State reports, AAUP, etc.), and the maintenance of a central file of all reports. In September and October of this year alone, there were over 50 reports involved. Each report may require action by several

different departments. Keeping track of these reports and assuring that deadline dates are met has become a time consuming task. Therefore, a Report Monitoring and Scheduling System was developed.

8.1 Operation of the System. When a report request is received, a secretary executes a program on an interactive terminal which will request her to type in the necessary monitoring and scheduling information -- report name, requesting agency, date due, and organization responsible for completion. From this point, the system produces two basic reports. The first one is a "scheduling" report.

The scheduling report may be run at any time for any month. It produces a calendar of all actions to be taken that month. All external reports required are listed, the date they are due, the requesting agency, the UTD department responsible for the report, date the report should be sent to the responsible department, and the date it is due back to Institutional Research for forwarding (and copying into a central file). A typical portion of this report is shown below.

MONTHLY REPORT FOR: NOVEMBER

REPORT	REQUESTING AGENCY	DATE DUE	RESP ORG	SEND TO RESP ORG BY	DUE BACK TO IIR
2300-2.3	HEGIS	NOVEMBER 1	REC	OCTOBER 1	OCTOBER 25
2300-2.5	HEGIS	NOVEMBER 30	REC	OCTOBER 30	NOVEMBER 23
2300-3	HEGIS	NOVEMBER 1	BUD	OCTOBER 1	OCTOBER 25

The other report produced is an End of Month Report. It reflects all actions which have taken place during the month, or were due to have taken place. A sample portion showing actions taken on one report is shown below.

END OF MONTH REPORT FOR: SEPTEMBER

REPORT: 2300-4

DATE DUE TO BUCR: SEPTEMBER 31

DATE SENT: SEPTEMBER 27

DATE DUE BACK TO INST RESEARCH: OCTOBER 24

DATE RECEIVED:

DATE DUE TO HEGIS: OCTOBER 31

DATE SENT:

8.2 Comments. This simple system has turned a confusing and time consuming task into a simple routine. Involvement of administrative heads, to expedite late reports, has been virtually eliminated. A centralized file and a centralized calendar has been created.

9. A TEXT, REPORT, AND QUESTIONNAIRE GENERATION SYSTEM

In the planning of undergraduate programs for the University, the academic administration wished to receive advice from many people all over the country knowledgeable in their field and who had special interest in undergraduate education. Therefore, the Delphi method, developed originally

by the Rand Corporation, was used. This method required many letters to be written; questionnaires to be created for each discipline surveyed; follow-up letters to be sent; and second-round questionnaires to be created and sent. A questionnaire generation system was developed to support this activity.

9.1 Description of the System. The system is designed so as to be easily used by a typist working at an interactive terminal. The first step is to enter the names and addresses of the panel members. These can be entered continuously and in any order. Each panelist and address will become a component of an APL file and hence have a unique number associated with it. This will become the panelist identification number. The "text-creation" program allows the body of the letter to be typed, corrections easily made, then run through an edit program to set margins, indentions, page numbering, etc.

- After these data are entered, no further input is entered. The "letter writing" program alphabetizes the panelist, prints the date, address, salutation, body of the letter, then stops until the next sheet of paper is inserted and the carriage returned. Envelopes can be also addressed, or for large mailing, labels printed from the APL file through the high speed printer.

Perhaps the only original part of the system is a general purpose program for creating questionnaires. The questions are printed in any desired space on the left side of the page and can be of varying length. The right-hand side then produces the desired format for the response; i.e., the question may ask for the left-hand statement to be rated as to its relevance

and importance to undergraduate education on a scale from 1 to 7.

9.2 Comments. This simple system was quickly written with the available resources and equipment. It got the job done quickly and effectively. Also, it was written in a general enough form to be used for a multitude of purposes.

10. BUDGET ANALYSIS

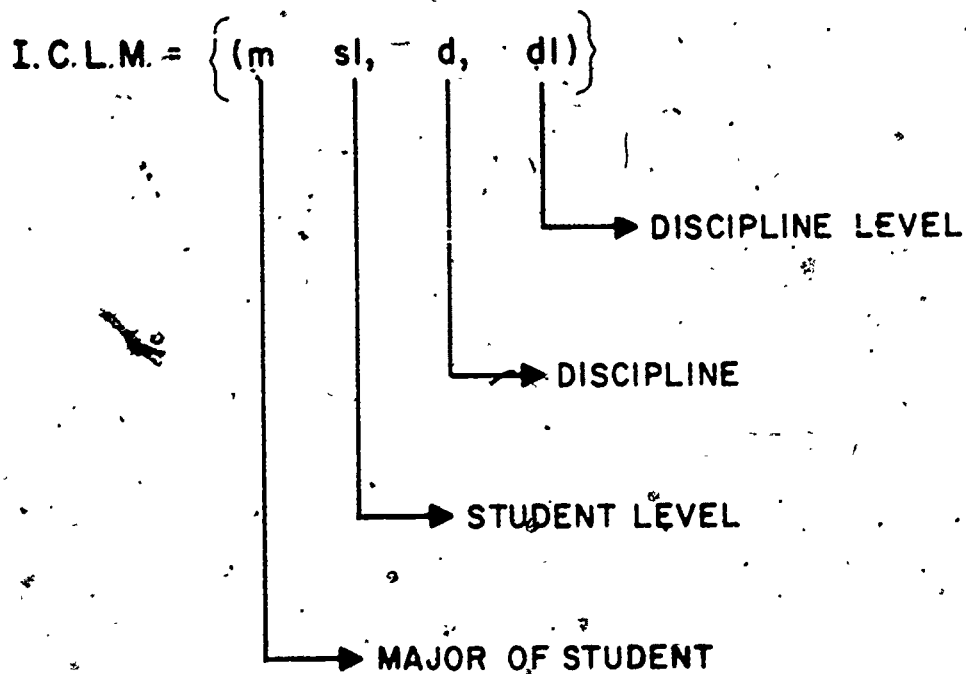
Numerous budget analysis systems were developed and used in studies for the Vice President for Business Affairs. However, budget analysis systems were also developed for Academic Affairs. For example, in planning for fiscal year 1974-75, the Vice President for Academic Affairs wanted to devise a system for the allocation of incentive funds among academic departments.

A deterministic model was devised. It consisted of three basic components: research assistants, organized research, and faculty salaries. In each component, the basis for calculating the amount of incentive funds to be awarded to a program was a linear equation based upon (1) sponsored research (contract and grant funds secured); (2) faculty salaries covered by sponsored research; and, (3) student assistant salaries provided by sponsored research. The model was run over 50 times, varying the equations and examining the results until a strategy was decided and agreed to. The system was interactive and therefore could be rapidly altered and run again.

11. CONCLUSION

By (1) operating under tough-minded and no-nonsense guidelines, and (2) utilizing effective tools such as interactive terminals and AFL, the Office of Institutional Research has managed to provide meaningful and effective assistance to the executive management of the institution. The information systems developed, when considered from a technical viewpoint,

have varied from pedestrian to sophisticated. The type of computerized system and degree of sophistication have been dictated only by what was needed to furnish the necessary management information. For Institutional Research, this is as it should be.

APPENDIX A**Example 1:**

(BIOLOGY, UD, STATISTICS, MAS) = 1.3 SCH.

i.e., A upper-division level student, majoring in Biology,
will average 1.3 SCH of Statistics at the Masters level.

EXAMPLE 2

Vol. II

	Discipline	UD	MAS	Ph.D.	
PGM A	A	10.5	3	6.4	B.A./B.S.
	B	5.5	9.1	11	
	C	3.7	11.4	0.8	
	D	6.9	5.3	1.9	
	E	12	10.2	5	
	A	3.5	10.4	11.8	MAT
	B	9.6	5.9	3.2	
	C	4.1	0.7	1.5	
	D	4.6	8.7	3.3	
	E	11.7	2.8	0.9	
	A	2.2	6.6	8.4	MAS
	B	2.3	5.8	10	
	C	2.1	0.1	11.5	
	D	6.1	0.2*	9	
	E	2	2.7	7.1	
	A	9.2	11.6	5.4	Ph.D.
	B	9.9	1.7	11.3	
	C	8.2	10.6	4.3	
	D	5.6	3.4	9.8	
	E	0.6	10.1	6.3	
PGM B	A	11.9	2.9	4	B.A./B.S.
	B	8.3	6.2	7.2	
	C	6.8	0.3	4.9	
	D	2.6	3.1	3.6	
	E	0.5	9.5	2.5	
	A	11.1	4.2	1.1	MAT
	B	1	11.2	10.8	
	C	3.9	1.2	4.8	
	D	1.4	4.4	8.6	
	E	9.7	0.4	2.4	
	A	8.8	10.9	5.2	MAS
	B	3.8	7	9.7	
	C	8.5	7.6	1.3	
	D	6.5	6.7	7.5	
	E	10.7	1.8	10.3	
	A	4.5	8.9	1.6	Ph.D.
	B	7.9	9.4	4.7	
	C	5.1	7.8	6	
	D	9.3	7.4	7.3	
	E	8.1	8	7.7	

* Example: A Masters level student, majoring in Program A, will average 2.7 SCH in discipline E at the Masters course level.

REFERENCES

- [1] S. C. Fallis, D. D. McElroy, and E. E. Payne, An On-Line MIS to Assist the Management of Facilities, Proceedings of the 1973 CAUSE Conference, December, 1973, 448-455.
- [2] Robert A. Huff, Overview of the Cost Estimation Model, NCHEMS, Boulder Colorado, April, 1971.
- [3] K. M. Hussain, A Resource Requirements Prediction Model (RRPM-1), Technical Report 20, NCHEMS, Boulder, Colorado, October, 1971.
- [4] Kenneth E. Iverson, A Programming Language, John Wiley and Sons, Inc., New York, 1962.
- [5] Leonard Gilman and Allen J. Rose, AFL, An Interactive Approach, Second Edition, John Wiley and Sons, Inc., New York, 1974.
- [6] J. R. Woolf, Space Factors and Space Utilization Values for Use in Meeting the Needs of Texas Colleges and Universities, Study Paper 12, Texas College and University System Coordinating Board, Austin, Texas, July, 1971.

OFF-CAMPUS GRADUATE STUDENT RECORDS:
THE ON-LINE SYSTEM SOLUTION

Paul H. Hoepner
Associate Dean
Graduate School
Virginia Polytechnic Institute and State University

Ronald R. Bauer
Systems Analyst
Graduate School
Virginia Polytechnic Institute and State University

1. Introduction

The last three decades have witnessed a tremendous explosion in enrollments at institutions of higher education. Increased demand, coupled with a healthy economy has produced a 600* per cent increase in undergraduate enrollments since 1945.

The post World War II period has also produced significant increases in graduate school enrollment. Between 1945 and 1974, the number of graduate students in the United States has doubled.** These changes have been prompted by demands in both the public and private sectors for education beyond a baccalaureate degree; with future job advancement and salaries often contingent upon earning a Master's degree.

Traditionally, a person holding a baccalaureate degree entered graduate school within a very few years after graduation. Furthermore, those individuals devoted their full-time efforts to earning a graduate degree in a relatively short period (18 months for a Master's degree; three and a half years for the Doctorate). Admissions procedures and academic policies were geared to the mode of a full-time, on-campus, graduate student.

The last decade has ushered in a new era in graduate education, and the mode of advanced degree earning has changed. Faced by increased employment requirements to further their education, but stymied by family responsibilities and communal ties, a growing number of individuals have been reluctant to "pull up roots" and make a full-time commitment in pursuit of a graduate education. Their clarion call has been for locally produced part-time

*The Carnegie Commission on Higher Education, New Students and New Places. McGraw-Hill Book Co., New York, October, 1971, p. 129.

**Ibid., p. 131.

graduate educational opportunities.

Cognizant of these changing needs and demands, Virginia's Land Grant University has responded to its educational mission by offering off-campus graduate degree programs throughout the Commonwealth. From a modest beginning of 200 students in the mid 1960's, off-campus enrollment now exceeds 2,000 active graduate students. In addition to offering degree programs in such concentrated population areas as Reston, Richmond and Dahlgren, courses are also offered at virtually any map coordinate displaying sufficient potential demand to assure a viable program. Virginia Tech currently offers between 100 and 150 off-campus classes each quarter.

Administrative problems are magnified by the fact that personal and professional demands placed on part-time graduate students make it difficult for them to maintain continuous registration. Consequently, several quarters may be skipped between enrollments. The Graduate School and the Registrar's Office are charged with the responsibility of maintaining a complete academic record of all active and inactive students.

During 1971, two clerks worked on a full-time basis to support the administration of off-campus programs. They maintained files for 3500 active and inactive students. All applications and registrations were hand processed. Roll sheets, grade sheets and grade reports were typewritten. It became increasingly obvious that a computerized off-campus student record system was necessary to cope with the deluge of paper. Thus, in September of 1972, a tape oriented system was installed to assist the record keeping process. This marked the beginning of the computerization of off-campus graduate student records.

2. Policy

Virginia Tech has attempted to respond to the increased demand for part-time graduate studies by adopting a liberal admissions policy for off-campus graduate programs. Students are permitted to enroll in one class prior to being formerly accepted to the Graduate School. This permissiveness is contingent upon the student initiating and completing the admissions process within the six weeks.

3. Current Prospective

The Graduate School administrative record keeping requirements have produced a complex set of data processing specifications. During the last five years, the off-campus program has grown from 2500 students (900 active) to 7000 students (2000 active). Furthermore, record size requirements have tripled from 300 data bytes to 1000 bytes per student to accomodate a complete academic and demographic file.

As noted above a student may actually begin graduate studies prior to being officially accepted. Without a formal application, a student's registration form generates a very inadequate data record. Recognizing this, the registration form was modified in 1972 to parallel an admissions application. From the population of all registrations received that Fall quarter, a tape file was constructed to generate a computerized graduate student data base. A tape system, which had been in service for on-campus records since 1965, was adopted to serve the off-campus needs.

A subsequent modification of the registration form by the Accounting Office resulted in a document void of demographic data elements. This precipitated a problem with the student who decided to attend class, but never initiated the admission application process. No tape record was constructed for persons in this situation, and for all practical purposes, their attendance was unknown.

3.1 Admissions

A flexible admissions policy requires responsive administrative procedures. An initial course registration form may or may not be accompanied by a completed application, transcripts and letters of recommendation. If all materials are received, a decision can be made to accept or reject the applicant. In the event that some materials are outstanding, the student must be advised that an application and/or other credentials are required and the file must be complete within six weeks so that a decision regarding admission can be made prior to the end of the quarter.

Applicants who do not submit a complete set of credentials within the prescribed time limit are advised that they will be graded on a pass-fail basis and the single course in which they are currently enrolled may not be used for graduate degree credit. Admission to graduate school is rejected for failure to fulfill the requirements for a complete application.

Rejected applicants are advised that they may not register in subsequent quarters. Despite being so notified, some students persist in attempting to register for additional courses. The administration of the Graduate School must have the capability of monitoring a student's status so that those who were rejected for academic reasons, or for failure to provide the required credentials, will have future registrations voided.

Once admission status has been determined, the decision is communicated to the applicant in writing. Currently, letters to rejected and accepted applicants are typewritten. Productivity considerations dictate automation, while public relations require a personalized communique. Specifications have been developed for a subsystem to transmit address data to a Mag-Card Selectric Typewriter in the Graduate Office via a telephone link from the Computing Center. Name, address, and degree information will be transmitted and recorded

on a magnetic card which is then processed by an MCST (Magnetic-Card II). Accepted and rejected applicants will be processed in batches. Each applicant will receive a personalized letter of acceptance or rejection generated as a result of the telephone interface between the Computing Center and the Graduate Office.

3.2 Admission Status Reporting

The magnetic tape record did not permit instantaneous retrieval, updating, and reporting of applicant admissions status. This produced administrative problems because it was difficult to review and communicate this information when needed.

In April, 1974, the off-campus tape file was loaded into an on-line IMS (Information Management System) record system then in use for on-campus students. This on-line interaction permitted instantaneous retrieval, verification and updating of a student's status.

A computer program was written to process admission records and construct an Admissions Status Report. This listing includes an applicant's name and address, academic data related to the undergraduate major field of study, institution where the bachelor's degree was earned, date undergraduate degree was conferred, undergraduate grade point average, graduate degree, graduate major, graduate grade point average, graduate institution, date graduate degree was conferred, and admissions test scores. For management purposes the status reports are organized according to campus locations within the Commonwealth.

Applicants are sorted alphabetically, by departments. Each report is divided into four parts:

applicants pending
applicants accepted
applicants withdrawn
applicants rejected

Departments receive applicant status reports semi-monthly.

3.2.1 Pending Report

The objective of the Pending Report is to display all graduate applicants whose admissions are pending review. This report serves as a vital communications link between the Graduate School and the Departmental Admissions Committee. It notes the reasons why an applicant's credentials are incomplete (e.g., missing transcripts or letters of recommendation). It also permits a department to concentrate its recruiting efforts on highly qualified students by flagging those with outstanding credentials. Enough information is available to permit personal follow-up and avoid the risk of losing a good student because one letter of recommendation has been inadvertently delayed.

3.2.2 Accepted, Rejected and Withdrawn Reports

These parts of the applicant status report are in the same format as the applicant pending section. Control breaks, by campus location and department, are also the same for all parts of the applicant status report.

The Withdrawn Report lists the names of applicants who have withdrawn their application for admission.

3.3 Registration

Prior to 1972, it was difficult to manually place the right student into the right class. The advent of the tape system significantly increased the reliability of this process.

Nonetheless, many problems continued to exist. The registration form was quite archaic. Although it satisfied the needs of the Accounting Office, it did not provide sufficient data to identify students who were not formerly accepted. Persons in this category were accounted for only when the professor brought it to the attention of the Graduate School. The registration form was subsequently re-designed to include the student's name, address, social security number, and telephone number. That is, enough demographic information was included to identify the student and make further communications possible.

3.4 Roll Sheets

Institutional requirements set a high priority on recording the registration of all students via an accurate class roll. It is imperative that class rolls include all students who have paid their fees and are attending class.

The revised registration form is used to create a skeleton student record. The skeleton record makes it possible to identify the student and assign him an admission status for inclusion on class rolls. This annotation may be communicated by the professor so that students who are walk-ins and have not made formal application to graduate school may take whatever remedial steps are necessary to complete the application.

3.5 Grade Sheets

During the final week of the academic quarter, the Registrar's Office produces a grade sheet for each off-campus course. A grade sheet is very similar to a class roll, except that it contains the names of

all persons who have completed the course and are eligible to receive a grade.

Prior to September, 1972, grade sheets were hand processed. Between the preparation of class rolls and grade sheets, a tremendous amount of clerical effort was required to produce an accurate grade document. It was necessary to process all late resignations, drops, adds, and student status changes against the most recent class roll. The modified tape system has materially aided the production of accurate grade sheets. Future plans include an on-line grade system expected to be available before January, 1976.

3.6 Grade Reports

At the end of each academic quarter student grade reports are printed. It is desirable to have all grade sheets returned from the class professor before student grade reports are generated. If several grade sheets are returned late, it becomes necessary to generate a second grade report for students who completed the course represented on the late grade sheet.

Applicants who have not been admitted to the Graduate School (rejected or pending) are not awarded the letter grade assigned by the professor. University policy requires that grade reports of these students display a pass or fail grade, whichever is appropriate, and a message explaining why the letter grade was not reported. These students are notified that the course will not carry graduate credit and any future course requests will not be accepted by the registrar.

Prior to September, 1972, all off-campus grade reports had to be typewritten. Needless to say, this required a tremendous amount of

intensive clerical effort to insure complete accuracy. The modified tape system has significantly enhanced the efficiency and efficacy of generating grade reports.

4. Plan of Study

Graduate School policy requires that all graduate students submit a plan of study for approval. Plans of study serve as a contract between the student and the University to indicate how each individual anticipates satisfying his advanced degree requirements. Each plan of study is reviewed to be sure that it satisfies all university requirements regarding minimum and maximum number of course hours at several levels, when and where courses will be completed, number of hours to be transferred from other institutions, and minimum number of hours taught by full-time Virginia Tech faculty. Plans of study are monitored to maintain academic integrity.

In the future, plans of study will be entered on the computer via an on-line IBM 3270 terminal. The computer will review these plans to determine if specific degree requirements are satisfied.

4.1 Management Uses

A computerized plan of study has another important advantage. It can aid in predicting the demand for future courses, by location, throughout the Commonwealth. Demand is a function of all students' degree requirements, minus completed courses. If plans of study are accurately maintained, management reports can be generated to aid in the allocation of faculty resources, classroom facilities, library support, and to assure students that courses will be available when and where needed.

far enough in advance to produce sound academic and institutional planning.

The aforementioned considerations produce significant logistical problems. University policy requires that at least 50% of all course work taken by an off-campus graduate student be taught by full-time Virginia Tech faculty. In order to satisfy this requirement, department heads must allocate faculty resources to meet their on-campus and off-campus instructional commitments. Logistical problems are also encountered in making travel arrangements between Blacksburg and the off-campus locations. Sometimes this distance is in excess of 250 miles and several locations are not readily accessible by air travel.

Some of the off-campus courses cannot be supported by on-campus professors. Therefore, it becomes necessary to screen and recruit qualified adjunct faculty to fill the void.

A viable graduate program requires access to adequate library facilities. To supplement locally available reference materials, the Virginia Tech library responds to faculty requests by shipping books and journals to the off-campus locations.

The problem of identifying suitable off-campus classroom facilities in appropriate locations is a never ending task. The most difficult task is finding classroom space in the more remote areas of the state. Classes have met in firehouses, church basements, high school rooms, etc.

Solving these dynamic logistical management problems can be aided by the use of the computer and forecasting course demand based on plans of study.

4.2 Progress Reporting

A computerized graduate plan of study has several additional advantages. Shortly after the termination of each academic quarter, the program of study data base may be interfaced with the current grade system to produce a hard copy itinerary containing all up-to-date revisions and grades earned. A copy of the program of study will be sent to the student in the form of a progress report. Copies will also be sent to the student's faculty advisor and the Dean of the Graduate School in an endeavor to monitor acceptable academic progress. Students who are in academic difficulty can be identified early and given proper counseling. Thus, these reports serve a vital role in the effective administration of all graduate programs.

5. Systems Development

In 1971, the Graduate Office received the first module of the new on-line IMS student record system. An admissions module was implemented employing an IBM 2260 video display terminal for on-line data entry and retrieval (later superseded by two IBM 3270's now in use). A group of reports were developed that depended on admissions data developed in on-line mode. The package included the following major reports:

1. Applicant Status - pending, accepted, rejected;
2. Curriculum, department and college enrollment reports;
3. State Council - summary of graduate students enrolled, by City and County;
4. Incomplete grade report;
5. Graduate Student Academic Status Report

Actual off-campus computer systems development was begun during the Summer of 1972 when the on-campus traditional Basic Student Record tape system was employed to process off-campus graduate student records. Although all hand processing operations were virtually eliminated, a number of problems were encountered attempting to apply a system designed for an on-campus environment to process off-campus graduate student records. Many problems originated from the unique policy that permitted an off-campus student to enter a class prior to being admitted to the Graduate School coupled with an inadequate registration form that did not contain sufficient demographic data to identify the student and permit administrative follow-up.

In April of 1973, the Graduate School hired a Systems Analyst to revamp the tape system and to assist in the design and implementation of the new on-line modules. His major responsibility was to address himself to the unique requirements of the off-campus record keeping process.

During April, 1974, the off-campus student record tape was loaded into the on-line student data base. A campus location code was assigned to each off-campus record for identification purposes. This marked the beginning of on-line student record processing. Status reports were then available for off-campus applicants.

Application data were entered directly from the Graduate Office using an IBM 3270 terminal thereby eliminating the need for coding and keypunching and the associated inaccuracies. A systems interface was designed, programmed and implemented to bridge the gap between the on-line admissions module and the roll and grade reporting tape system. The tape system is still employed to generate class rolls, grade sheets and administrative reports.

6. Future Plans

The development of the student record system is by no means complete. Future plans include the implementation of a number of additional modules.

6.1 Optically Scanned Course Requests

An on-line registration system has been implemented for on-campus student record processing. There are two methods of data input. One, an optically scanned course request form is batch processed and then run through a class scheduling algorithm. Two, class drops and adds are processed via a 3270 on-line terminal.

Within the next year, all off-campus registrations will be processed in the same fashion. A pre-printed optically scannable form has been designed to fit a standard business size envelope. This form will be mailed to all active off-campus students. Completed forms will be returned to Blacksburg for computer batch processing.

Fees may be returned by mail and must be paid at least two weeks prior to the beginning of the quarter. Students who fail to meet the payment deadline will have their course registrations purged from the system.

6.2 On-Line Grade Processing

During the Winter of 1975, an on-line grade system will be installed on-campus. Machine readable grade sheets will be prepared for all classes. Grade sheets will then be optically processed similar to the course requests. Grades will be appended to each student's academic record. A grade report will be printed summarizing each student's

performance for the current quarter as well as accumulated grade point average. This system will also process off-campus grades.

6.3 Dial-Up Terminals

Several areas in the State of Virginia have high concentrations of potential graduate students, particularly in the metropolitan regions of Washington, D.C., Richmond, and Norfolk. The Graduate School, in conjunction with the Virginia Tech Computing Center, is studying the feasibility of using portable dial-up computer terminals at remote locations during registration. This will aid faculty advisors by permitting them to review current academic records for use in student consultation.

6.4 Inactive Records

A classical problem with all on-line systems is the high cost of storing inactive records. Based on past experience, approximately three out of every four off-campus records are identified as inactive students. That is, three out of every four records are not referred to or updated at least once each quarter. Because the population of active off-campus graduate students is very dynamic, inactive records must be removed from and recalled to the on-line environment as efficiently as possible. Magnetic tapes are used for off-line storage purposes.

Students who have not registered for at least one course during the current academic year will have their records removed from the on-line system. An on-line to off-line migration report is created to serve as an audit trail for inactive student records. This report contains sufficient information to locate inactive records should the need arise.

6.5 On-Campus Remote Data Retrieval

At present, each of the seven colleges has at least one IBM 3270 video display terminal. Keyed messages from these terminals are transmitted to an IBM 370/168 computer. Student records may be transmitted back to the college terminals on a need to know basis. No updating is performed at the college terminals except for registration transactions related to dropping or adding courses.

The use of remote terminals requires a reliable security system. Data security within Virginia Tech IMS information system is of two types. First, unique transaction codes and/or passwords are associated with each terminal. This is accomplished within the IMS software. Second, security may be controlled by the applications program. That is, not permitting one college and/or department to retrieve information for students enrolled in another college or department because of restrictions on a need to know basis. Once the data retrieval systems achieve the anticipated level of sophistication, on-line review of off-campus records will significantly enhance the efficacy of student advising.

7. Closing Statement

The most difficult problem associated with the development of the on-line student record system at Virginia Tech is maintaining a constant awareness of the unique data processing requirements of the off-campus environment. Our ultimate objective is to produce one record system capable of handling all students, regardless of type. This requires in-depth analysis and total understanding regarding the three distinct populations of students: undergraduates, on-campus and off-campus graduates.

Systems development is an interactive and iterative process. In our view, successful development effort can not be accomplished without constant communications between the user and the developer. The dynamics of a complex institution make it impossible to cast the total systems development specifications in bronze, a priori. We have chosen a modular development, with the constant view toward modifying the system to satisfy ever changing student record keeping requirements. We have attempted to maintain a constant vigilance and an awareness of new ideas and changing technology to guarantee maximum productivity from the allocation of scarce resources in the production and maintenance of on-line computer systems at Virginia Tech.

8. Corollary

One of the major lessons to be learned from our experience is the critical need to properly interface the data processing requirements of the administrator and the technical expertise of the Systems Analyst. Typically, the busy administrator has a reasonably but not necessarily completely defined set of needs, and only a nodding acquaintance with data processing. The analyst possesses the technical competence, but lacks first hand knowledge of unique administrative problems and requirements. We believe that casual periodic communications between administrator and developer may not produce a satisfactory system capable of encompassing the needs of a complex dynamic environment. What is needed, in our opinion, is a daily, shirtsleeve working arrangement, that meets the needs of the user, rather than simply satisfies the esoteric self-satisfying professional requirements of the producer.

THE REGGIE SYSTEM

Dr. Frederick A. Kundell
Associate Academic Dean
Salisbury State College

THE REGGIE SYSTEM

Frederick A. Kundell, Associate Academic Dean
Salisbury State College
Salisbury, Maryland 21801

Registration, because of the complex and capacious amount of information which must be stored, sorted, and analyzed, has been a prime target for computerization. The REGGIE system of programs was designed to facilitate the registration process by minimizing the manual effort and time involved while maintaining the decision making potential. The system is composed of programs which were designed to aid the registrar's office from the preparation of a conflict free course schedule to, when finished, the analysis of final grades. The system includes a sectioning algorithm which the author feels is superior both in run-time and accuracy to any other in existence. Finally, both the input and output were designed to be easily understood by people who do not have a data processing background.

1. INTRODUCTION

Registration is a necessary component in and evil of our present collegiate educational system. Its goal is the documented assignment of students to courses in accordance with the conditions imposed by the physical plant and academic community. The complexity of this procedure varies with the size of the student body and the procedure employed.

The REGGIE system of programs was designed to facilitate the registration process by minimizing the manual effort and time involved while maintaining the decision making potential. The remainder of this paper will be directed toward an overview of the system.

2. SYSTEM CONFIGURATION*

The REGGIE system is composed of a set of nucleus subroutines, which remain

* Based on X-RAY 67, PROGRAM SYSTEM FOR X-RAY CRYSTALLOGRAPHY, J. M. Stewart, F. A. Kundell, et al, Technical Report 67-58, Computer Science Center, University of Maryland

in core at all times, plus the system programs, which reside as overlays. The nucleus subroutines handle all input-output and act as a system monitor. This design yields a very flexible system in which the program sequence is controlled by the input card stream. While the programs were written to work as an integrated system, they can easily be separated for independent use.

2.1 CODING. The system is written in Univac 1100 series FORTRAN V. Each subroutine is headed with a brief introduction which states its purpose and position in the system. In addition, the code is broken into logical blocks. Each of these blocks is introduced with a fairly complete description. Within the blocks each logical operation is headed with an explanatory comment (APPENDIX I).

3. SCHEDULE FILE FORMAT

The system, as it is now constituted, utilizes a sequential data file composed of ten physical records. Each physical record is composed of from one to 'N' logical records. The size of the logical record is set in accordance with the software specifications of the machine (eg. on the Univac 1106 there are 255 usable words per file record). The ten physical records are assigned as follows:

1. File history and key record
2. Instructor record
3. Room record
4. Student information record
5. Course request - course assignment record
6. Course schedule record
7. Abbreviated course schedule record
8. Grade roster record
9. Not assigned
10. Acts as a system end of file

Physical record one contains the file history and key. The file can only be read by the system if a duplicate key is submitted at the beginning of the job stream. The first file access causes the file history to be printed (APPENDIX II, Sample Listing 2).

Additional physical records may easily be added as the need arises. Furthermore, by modifying the nucleus routines, the desired information could be retrieved randomly from a number of different files. A dump of the 'REGGIE TEST DECK' schedule files appears in APPENDIX IV.

4. MASTER COURSE SCHEDULE PREPARATION

A well-planned master course schedule can vastly simplify subsequent operations. A number of programs have been included in the REGGIE system to facilitate this process. It is generally recognized that the manual maintenance of parallel files is difficult, if not impossible. Consequently, the REGGIE system was designed to read the same course schedule as is distributed to the students. In fact, the system prepares the master listing for publication (APPENDIX II, Sample Listings 3 and 4). This schedule, having been thoroughly checked by the system, has a minimum of clerical and scheduling errors.

4.1 THE LOAD-SCHEDULE PROGRAM. The schedule file is initialized via the LOAD-SCHEDULE program. It creates physical records one, two, three, six and seven. The remaining records are initialized as 'dummy' records. The first deck read is the instructor deck. This deck contains one card per instructor and is normally loaded in alphabetic order (APPENDIX II, Sample Listing 1). The second deck contains room information. The system utilizes the master room deck prescribed by the Board of Trustees of the Maryland State Colleges*. It should be noted that both the instructor and room listings have been well received and utilized on campus (APPENDIX II, Sample Listings 5 and 6).

The master course schedule is loaded in two steps. The first, the LAB-LIM deck, contains abbreviated course titles, used in the preparation of the permanent record (LABEL), the class limit and type, the sectioning group to which the section belongs (blank if it is to be assigned by the system), and

* A variation of the format listed in "Higher Education Facilities Classification and Inventory Procedures Manual" published by the Maryland Council for Higher Education.

whether the course is being taught for a set fee (part-time or overload). The second deck contains the course schedule in a format similar to the published form.

The LOAD-SCHEDULE program checks the submitted course schedule to make sure that the information is complete. The items checked are the course number, title, semester hours credit, meeting time, abbreviated title (label), and the limit. In addition, it checks the assigned room against the master room list and the instructor against the instructor list. If an error is detected, it is noted to the right of the master schedule listing (APPENDIX II, Sample Listing 7). Subsequently, the class limit is checked against the room capacity. If the capacity is exceeded, this fact is noted on the LAB-LIM listing (APPENDIX II, Sample Listing 8).

As the course schedule is being read, a condensed schedule is developed in core. This schedule contains abbreviated instructor and room information plus for each course, the department abbreviation, the course number, the class limit, a blank word for enrollment, the group symbol for sectioning, the semester hours credit, and pointers to the instructor, room and time information.

The clock hours, as they appear in the schedule of classes, are translated into a readily usable form by the LOAD-SCHEDULE program. The stored time is composed of three segments of one, four, and three decimal positions respectively.

.X / XXXX / XXX

Segment 1 The End Flag

This character is zero for all but the last meeting time of a section. It is unity for the last.

Segment 2 The Position Pointer

The position pointer indicates the starting time of the class meeting. The week, Monday through Saturday, from 6AM to midnight is broken up into five-minute intervals as follows:

Time	M	T	W	R	F	S
6:00	1	2	3	4	5	6
6:05	7	8	9	10	11	12
6:10	13	14	15	16	17	18
6:15	19	20	21	22	23	24
6:20	25	26	27	28	29	30
6:25	31	32	33	34	35	36
6:30	37	38	39	40	41	42
6:35	43	44	45	46	47	48
etc.						

A position pointer for Monday at 6AM would be 0001. One for Thursday at 6:35AM would be 0046.

Segment 3 The Duration Counter

The duration counter is the number of five-minute intervals in a class meeting.

Examples

MWF 6 00001010, 00003010, 10005010
 TR 6:15-6:35 00020004, 10022004

The following are examples of acceptable time notation:

MWF 9 (50 min. class)	T 5:00-6PM
S 9:00-11:45	S 9-11:45
MWF 5:6:15 PM	TR 1:30-2:45
DAILY 9	DAILY 10:30-11:45
MWRF 10	MTWRF 1
TBS	TBA

The condensed schedule is loaded onto physical record seven as a core dump.

This record is used by numerous programs in the system.

It should be noted that the LOAD-SCHEDULE program can be run in an update mode. In this mode, either entire decks can be replaced, or they can be corrected using the DELETE and/or INSERT card options.

4.2 THE ROOM-CHK AND PROF-CHK PROGRAMS. The ROOM-CHK program was designed to analyze classroom utilization. It prepares a room schedule for each room referred to in the master course schedule. Each class is indicated in its proper time slot with the course number, instructor, and class limit noted. After the student course requests have been loaded, the class limit can be replaced by the actual number of requests for the given course. If conflicts

occur, they are noted and the conflicting courses indicated to the right of the time grid. All classes assigned to the given room are considered. If there was a time error or if the course time was not specified (TBS or TBA), this information is noted. In addition, the room utilization is calculated and printed.

The ROOM-CHK program can be used to list all rooms referred to in the master course schedule, those in a given building, specific rooms, or a range of rooms. In addition, if specified, it will only list those rooms in which conflict occurs (APPENDIX II, Sample Listings 10 through 13).

The PROF-CHK program is comparable to ROOM-CHK (APPENDIX II, Sample Listings 14 and 15). It prepares the schedule for each instructor listed in the master course schedule. The user also have the option to list schedules for specific instructors or a range of instructors. Again the PROF-CHK program can be run in the conflict mode.

During master course schedule preparation, we normally run the LOAD-SCHEDULE program, in either the a priori or update mode, plus a complete ROOM-CHK and PROF-CHK, listing only those instructors who have conflicts. Using this procedure we can develop a conflict-free schedule in a matter of hours. Normally we devote a couple of days to the process.

At the beginning of each semester a full ROOM-CHK and PROF-CHK listing is prepared and distributed to all administrative and departmental offices on campus. Thus anyone who wishes to contact an instructor has his schedule available. The ROOM-CHK listing is used extensively by the maintenance and instructional resources departments. It is important for them to know when classrooms are free for cleaning, repair, or for setting up audio-visual equipment.

It would be a savings of both time and paper if the ROOM-CHK and PROF-CHK programs could be run on-line using a CRT. We plan to make this change as soon as the hardware becomes available on campus.

5.. REGISTRATION

The primary reason for the development of the REGGIE system was the testing of a new sectioning algorithm*. While the schedule preparation programs have become extremely important in their own right, their development was dictated by the need for physical record seven, the condensed course schedule. In this section, the implementation of "An Algorithm For Computer Registration" shall be discussed along with the results of the Salisbury State College fall registration.

5.1 THE LOAD-NAME PROGRAM. The LOAD-NAME program loads student information into physical record four of the schedule file (APPENDIX III, Sample Listing 1). At the present time we only load the student name and social security number. This record should be expanded to include other useful information such as the major, student classification, etc. The major is of particular importance if a cross-over study program is to be added to the system.

Since alphabetic student listings are required throughout the system, the student names are loaded into physical record four in alphabetic order. When read out by other programs, an alpha sequence number is appended to the record. In most cases, the student name - social security number cards are loaded in alphabetic order. For this reason a sort algorithm was desired which could recognize this fact and thus avoid a time-consuming and costly sort. While the algorithm developed has probably been used countless times before, no one in our shop had run across it. For this reason, I have added a brief description in APPENDIX VI.

5.2 THE TALLY PROGRAM. The initial phase of the sectioning algorithm is a tally of course requests. This tally is stored in physical record seven and is used in developing the status indicator in the sectioning program. In

* Frederick A. Kundell, An Algorithm For Computer Registration, College and University, Vol. 48, Winter, 1973, pp. 87-89.

addition to this, the TALLY program serves several other critical functions in the system.

First, the TALLY program builds the student course request record, physical record five. The program is designed to run either in an a priori mode from cards or from a previously prepared data file, or both.

Second, the program checks the student requests against the student information record, physical record four. If present, the student's name and the alpha sequence number is appended to the course requests. If not present, the name is set to blanks and the alpha sequence number set to zero. In addition, an entry is made in the TALLY diagnostic report (APPENDIX III, Sample Listing 2).

Third, the student course requests are checked against the master course schedule. If the course, or section, is not being offered, the fact is noted in the diagnostic report (APPENDIX III, Sample Listing 2).

Finally, the program lists the student requests by section and student classification. This listing is examined with the department chairman to see if modifications to the master course schedule are required (APPENDIX III, Sample Listing 3). A copy of the TALLY listing is also sent to the college book store.

Two additional reports are generated by the TALLY program. The first is an analysis by course and sectioning group (APPENDIX III, Sample Listing 4). The second report gives the overall TALLY statistics (APPENDIX III, Sample Listing 5).

5.2 THE SECTIONING PROGRAM. The sectioning program begins by reading physical record seven, the condensed schedule. This schedule now contains the number of course requests as calculated by TALLY. A status indicator for each section in the schedule is calculated by subtracting the class limit from the number of student requests. If the status indicator is zero, the class is full (i.e., all available seats have been taken). A negative status indicator is

numerically equivalent to the number of unsubscribed seats in the section and a positive status indicator is numerically equivalent to the over subscription. Subsequently, the status indicators for all sections in a given course and group are summed. This number is referred to as the course status indicator or CSI. The CSI has the same numerical significance as the status indicator, but relative to the entire course, or group within the course.

The student course records are now read from physical record five. The status indicator for each requested section is first checked. If it is zero or negative, the student is enrolled in the section. If it is positive, the section is oversubscribed. In such a case, an alternative section is sought if the CSI is not positive. If the CSI is positive, the course is oversubscribed and the student is not enrolled in the course. In such a case both the status indicator and CSI are decreased by one. If the CSI is zero or negative, an alternative section is sought. The alternative section must have a negative status indicator and be compatible with the remainder of the student's schedule. If an alternative section is found, the student is enrolled in it and its status indicator is increased by one. At the same time, the status indicator of the requested section must be decreased by one. If an alternative section is not found, the student is tentatively enrolled in the requested section.

After all student requests have been considered, the request record is copied onto the new schedule file. Prior to the actual copying, all status indicators are checked. Status indicators which are still positive are noted. In the copying phase the course assignments are scanned for oversubscribed sections. When found, the student is dropped from the section and the status indicator decreased by one. When all status indicators become zero or negative, the remaining course assignments are simply copied across.

The SECTION program prepares a statistical page as its sole output (APPENDIX III, Sample Listing 6). In normal operations the SECTION run is followed by a second TALLY run (APPENDIX III, Sample Listing 7).

5.4 THE SCHEDULE PROGRAM. Three programs play a primary role in the initial registration process. The first two, TALLY and SECTION, have already been considered. The last is the student schedule preparation program "Schedule" (APPENDIX III, Sample Listing 8). The schedule card contains two message fields. The first is printed on all schedule cards. The second is printed for only students with incomplete schedules. The text of both fields is supplied by the user.

5.5 REGISTRATION STATUS FLAG. When the student course requests are loaded by the TALLY program, a status flag is inserted into the record. TALLY initializes this flag to zero. The SECTION program considers only those records with a zero status flag and in turn converts the flag to minus one. The SCHEDULE program prints schedules for only those students with a status flag equal to minus one or zero and in turn converts the status flag to plus one.

The registrar, in general, has two options in the registration process. First, the registrar can choose to honor all requests from a certain group of students by by-passing the sectioning program (i.e., by running only TALLY and SECTION). In this way, the status flag which was initialized to zero by TALLY is converted to one by the SECTION program and the resultant schedules will not be affected by subsequent operations. The second procedure includes the SECTION program as discussed above. Once a batch of requests has been processed, the results will not be affected by the processing of subsequent batches.

5.6 GROUP SECTION DESIGNATION. In most courses, all sections are equivalent. However, this need not always be the case. As an example, a certain section of introductory English composition may be set aside for nursing students so as to be compatible with the remainder of their schedule. In such cases, the system must not shift other students into this restricted enrollment system. This grouping is allowed for by the group designation option on the LAB-LIM

card. If the group field is blank, the system assigns group designators, (e.g. D1 for day lecture, L1 for day laboratory, E1 for evening lecture, EL for evening laboratory, etc.). If a group designator is supplied, it overrides the system designation. In the sectioning process, students will only be shifted to an alternative section if it is the same course and has the same group designation.

5.7 STUDENT AND COURSE PRIORITIES. The registrar establishes student priority by the processing sequence and the ordering of requests. As discussed above, the registrar may choose to honor all requests by simply skipping the SECTION program. In the sectioning process, the requests at the beginning of the queue are most likely to be changed or refused. Therefore, the request should be submitted in increasing priority order (i.e., highest priority last per run).

The student may rank his/her requests. The system treats the course requests from a given student in decreasing priority order (i.e., the first request has the highest priority).

5.8 RESULTS OF THE FALL 1974 SSC REGISTRATION. Four separate registration batches were run for the fall 1974 registration at Salisbury State College. The results were as follows:

BATCH 1	Run Sequence - TALLY, SECTION, SCHEDULE
	Number of Students 2,005
	Courses Requested approx. 11,500
	Complete Schedules 1,735 (86.5%)
	Incomplete Schedules 270
	Miscellaneous* 243
	Program Failures 27 (1.35%)
	CPU Time 2.5 minutes
	Computer Time 4.0 minutes

* Oversubscription, nonexistent courses, or a conflict in the requested courses which could not be resolved.

BATCH 2 & 3

Run Sequence - TALLY, SECTION, SCHEDULE	
Number of Students	300
Course Requests	approx. 1,500
Complete Schedules	270 (90%)
Incomplete Schedules	30
Miscellaneous	30
Program Failures	0

BATCH 4

Run Sequence - TALLY, SCHEDULE	
Number of Students	approx. 710
Complete Schedules	710

Composite

Run Sequence - TALLY, SECTION, SCHEDULE	
Number of Students	2,305
Complete Schedules	2,005 (87%)
Program Failure	27 (1.2%)

Overall

Number of Students	3,015
Complete Schedules	2,715 (90%)

5.9 IMPROVEMENT TO SECTIONING PROGRAM. It was noted that the number of incomplete schedules in the miscellaneous category was higher than expected. A separated run was made with approximately 1,400 requests, primarily from freshmen and sophomores, in which the initial requests were analyzed. To our great surprise, approximately 18% of the initial requests contained course conflicts. The program, in such a case, would accept the first conflicting course and search for an alternative to the second. This process rectified most conflicts. However, better results could be expected if both courses were considered for alternate sectioning. The program has now been expanded to reschedule all cases where an incomplete schedule results. In the second pass, the courses are considered in reverse order. The best result is accepted.

5.10 THE LIST-STUDENT PROGRAM. The LIST-STUDENT program prepares a list of all registered students. This list can be in either alphabetic or student number order (APPENDIX III, Sample Listing 9).

5.11 THE ROSTER PROGRAM. The ROSTER will prepare four different rosters.

They are:

1. Class roster on stock paper
2. Grade roster on stock paper
3. Class roster on printed form
4. Grade roster on printed form

Whenever a change in forms or spacing is required, the program goes into a countdown to enable the operator time to set the printer (APPENDIX III, Sample Listing 10). This listing shows the echo input specification dump. A dump of this type is prepared by all programs which have options. A sample class roster appears in APPENDIX III, Sample Listing 11.

5.12 THE LIST-ROSTER PROGRAM. There are numerous occasions when it would be convenient to have a list of all sections being offered during a given semester. This listing is prepared by the LIST-ROSTER program (APPENDIX III, Sample Listing 12).

5.13 THE DROP-ADD PROGRAM. The DROP-ADD program for student schedule modification is now being written. A sample listing is not available at this time.

6. REGISTRATION ANALYSIS AND DATA MAINTENANCE

At this time, there are only three programs in this section of the system. They are DUMP, COPY, AND PURGE. The DUMP program prepares a listing of any or all of the physical record (APPENDIX IV). The COPY program copies the schedule file onto a secondary unit. This is normally done at the end of each semester, so that a permanent record can be saved. The PURGE program replaces any specified physical record with a dummy record.

Plans call for a number of analyses programs plus programs to handle grade information.

7. DATA PREPARATION

A listing of the data cards used for this sample listings appears in APPENDIX V. While the system writeup is not yet complete, it is available on request.

8. CONCLUSION

It is my sincere hope that this paper has presented information which you will find useful. We do not have all the answers, but we are continuously striving to improve the registration process. The REGGIE system illustrates an integrated system approach using a single data file. Among the system's

strengths is hopefully a very readable printout and a very simple operating procedure. The major strength is the sectioning algorithm, which I feel is superior both in run-time and accuracy to any other in existence.

EDITOR'S NOTE: A few output samples of the REGGIE System follows. A full set of appendices is available from the author on request.

REGGIE-LOAD-SCH RUN TIME 11:11 11/27/74 REGGIE TEST DECK
 SEQ COURSE NO. TITLE SH DAYS & HOURS ROOM INSTRUCTOR LIMIT POSSIBLE ERRORS

1* ART

2* 12 101 01 PRINCIPLES OF ART 3 MW 11:00-1:25 BH 283 BAROQUE 20
 3* 12 101 02 TR 11:00-1:25 BH 283 BAROQUE 20

6* BIOLOGY

7* 14 101 01 GENERAL BIOLOGY 4 MWF 9 BH 251 FERN 50
 8* LAB 11 M 3:00-4:50 BH 130 FERN 15
 9* LAB 12 W 10:00-11:50 BH 130 PLANT 15
 10* LAB 13 F 11:00-12:50 BH 130 FERN 15
 11* LAB 14 T 12:00-1:50 BH 130 PLANT 15

14* ENGLISH

15* 30 101 01 PRINCIPLES OF COMPOSITION 3 MWF 8 BH 248 THOREAU 10
 16* 30 101 02 TR 2-3:15 BH 248 LUNGUSTIC 10
 17* 30 101 03 MWF 12 BH 251 THOREAU 10
 18* 30 101 04 MWF 2 BH 251 LUNGUSTIC 10
 19* 30 101 05 MWF 4 BH 248 LUNGUSTIC 10
 20* 30 101 06 TR 7-8:20 PM BH 248 MOONLITE 10 INSTR
 21* 30 101 07 TR 11-12:15 BH 248 THOREAU 10

24* FRENCH

32 101 01 ELEMENTARY FRENCH

PARLEY 12

APPENDIX II

Sample Listing 7
 LOAD SCHEDULE
 Master Schedule Listing With
 Possible Errors Noted

25*
26*
27*

REGGIE-PROF-CHK		PUR. TIME 11:18 10/18/74		REGGIE TEST DECK			
NO: DAY		TUESDAY	WEDNESDAY	THURSDAY	FRIDAY	SATURDAY	
8:00	I	I	I	I	I	I	I
9:00	I ENCL 101 02	I	I ENGL 101 02	I	I ENGL 101 02	I	I
	I // BH 248 //	I	I // BH 248 //	I	I // BH 248 //	I	I
	I // BH 248 //	I	I // BH 248 //	I	I // BH 248 //	I	I
	I // BH 248 //	I	I // BH 248 //	I	I // BH 248 //	I	I
	I // BH 248 //	I	I // BH 248 //	I	I // BH 248 //	I	I
10:00	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
11:00	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
12:00	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
1:00	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
2:00	I ENCL 101 04	I	I ENGL 101 04	I	I ENGL 101 04	I	I
	I // BH 251 //	I	I // BH 251 //	I	I // BH 251 //	I	I
	I // BH 251 //	I	I // BH 251 //	I	I // BH 251 //	I	I
	I // BH 251 //	I	I // BH 251 //	I	I // BH 251 //	I	I
	I // BH 251 //	I	I // BH 251 //	I	I // BH 251 //	I	I
	I // BH 251 //	I	I // BH 251 //	I	I // BH 251 //	I	I
3:00	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
4:00	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
5:00	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
6:00	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
7:00	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
	I	I	I	I	I	I	I
8:00	I	I	I	I	I	I	I

APPENDIX II

Sample Listing 15

PROF-CHK Listing

With Conflict And Time Error
(from earlier run)

INSTRUCTOR LUNGUISTIC

DEPARTMENT ENG ASSISTANT

CONFLICT
ENGL 101 06 ENGL 101 05

TEACHING LOAD HRS.

LECTURE 12.00
LABORATORY .00
TBS .00
TOTAL 12.00

CREDIT HOURS TAUGHT 15.00

COURSES NOT SHOWN

ENGL 101 08 3.0 ERROR

REGGIE-SECTION

RUN TIME 11:36 11/27/74

REGGIE TEST DECK

SECTIONING STATISTICS

TOTAL NUMBER OF STUDENTS	51	NUMBER	PERCENTAGE
STUDENTS WITH COMPLETE SCHEDULES	41	80.39	
STUDENTS WITH INCOMPLETE SCHEDULES			
OVER SUBSCRIPTION OR NON-EXISTANT COURSES	10	19.61	
SECTIONING CONFLICTS	0	.00	
TOTAL	10	19.61	
TOTAL NUMBER OF REQUESTS	314		
REQUESTS FULFILLED	304	96.82	
REQUESTS NOT FULFILLED			
OVER SUBSCRIPTION OR NON-EXISTANT COURSES	10	3.18	
SECTIONING CONFLICTS	0	.00	
TOTAL	10	3.18	

APPENDIX III

Sample Listing 6
SECTION Results

ABLE I. M.
900-10-0008 (UG)

SALISBURY STATE COLLEGE

11/27/74 REGGIE

COURSE SCHEDULE
FALL SEMESTER
1974-1975

DEPT	COURSE NO.	SH	DAYS & HOURS	ROOM
ENGL	30 101 05	3	MWF 4	BH 248
FREN	32 101 01	3	MWF 10	BH 251
HIST	40 101 02	3	MWF 11	BH 251
MATH	50 101 02	3	MWF 2	BH 248
PHED	60 101 02	1	WF 8	FIELD
PSYC	70 101 02	3	TR 1-2:15	BH 251

IF YOU HAVE ANY
QUESTIONS, PLEASE
CONTACT THE OFFICE
OF THE REGISTRAR

TOTAL HOURS SCHEDULED 16.0

TOTAL HOURS REQUESTED 16.0

BLITZ SPEEDY
900-10-0048 (UG)

SALISBURY STATE COLLEGE

11/27/74 REGGIE

COURSE SCHEDULE
FALL SEMESTER
1974-1975

DEPT	COURSE NO.	SH	DAYS & HOURS	ROOM
ART	12 101 02	3	TR 11:00-1:25	BH 283
ENGL	30 101 01	3	MWF 8	BH 248
MUSC	54 101 01	3	MWF 1	BH 251
PSYC	70 101 03	3	MWF 11	BH 248
HIST	40 101 03	3	MWF 3	BH 248

IF YOU HAVE ANY
QUESTIONS, PLEASE
CONTACT THE OFFICE
OF THE REGISTRAR

TOTAL HOURS SCHEDULED 15.0

NOTE YOU ARE NOT REGISTERED IN THE FOLLOWING
FREN 32 101 01 3

SINCE YOU DID NOT
GET A COMPLETE
SCHEDULE,
ADJUSTMENTS MAY BE
MADE AT PRIORITY
DROP-ADD.

TOTAL HOURS REQUESTED 18.0

FOOT WRIGHT
900-10-0013 (UG)

SALISBURY STATE COLLEGE

11/27/74 REGGIE

COURSE SCHEDULE
FALL SEMESTER
1974-1975

DEPT	COURSE NO.	SH	DAYS & HOURS	ROOM
BIOL	14 101 01	4	MWF 9	BH 251
ENGL	30 101 04	3	MWF 2	BH 251
HIST	40 101 01	3	MWF 10	BH 248
MATH	50 101 03	3	TR 9:30-10:45	BH 251
PHED	60 101 02	1	WF 8	FIELD

IF YOU HAVE ANY
QUESTIONS, PLEASE
CONTACT THE OFFICE
OF THE REGISTRAR

TOTAL HOURS SCHEDULED 14.0

NOTE YOU ARE NOT REGISTERED IN THE FOLLOWING
14 101 03 SECTION NOT IN SCHEDULE

SINCE YOU DID NOT
GET A COMPLETE
SCHEDULE,
ADJUSTMENTS MAY BE
MADE AT PRIORITY
DROP-ADD.

TOTAL HOURS REQUESTED 14.0

APPENDIX III

Sample Listing 8
Student Schedule Cards

COMMITTEE APPROACH TO USER
INVOLVEMENT IN THE ESTABLISHMENT
OF A COLLEGE INFORMATION SYSTEM

Barbara F. Medina
Director
Data Processing
Herbert H. Lehman College

Anthony G. Picciano
Program Manager
Data Processing
Herbert H. Lehman College

THE COMMITTEE APPROACH TO USER INVOLVEMENT IN THE ESTABLISHMENT OF A COLLEGE INFORMATION SYSTEM

Barbara F. Medina, Director of Data Processing, Herbert H. Lehman College, Bronx, N.Y.
Anthony G. Picciano, Program Manager, Herbert H. Lehman College, Bronx, N.Y.

1. INTRODUCTION

In attempting to develop and implement a Student Information System for Lehman College, we had to face several major problems. The system was to use an integrated data base approach that would record and report on a student's records from application until a termination of active interface with the college. Therefore, it was necessary that several offices reporting along separate organization lines maintain and share the same files. Decisions relating to college policy on registration, grade report distribution, financial aid information, accessibility and security of data had to be made by several upper administrators of the college. The limitations and advantages of computerized records had to be explained to all potential administrative users of the system so that the system could be designed as a useful management tool for both upper management and the operational management. Since most of the users were unfamiliar with both using computerized files and with the functions of other offices in the college, which would depend on the files for information, it was necessary to institute some type of educational plan.

Initially, the approach we used to try and solve these problems was: the Data Processing Department would write proposals and request comments and suggestions from users. These proposals included the use of software developed at other colleges. This proved to be a slow and often unfruitful approach because the users frequently did not understand the implications of the proposals. The solution that provided the proper interaction between users and the data processing project manager was the formation of a working committee. The committee

members now include a representative of all potential user offices and is chaired by the data processing project manager. It has committed itself to meeting once a week for at least two hours each week until all problems are ironed out and the system is fully operable. Minutes are recorded at each meeting by the project manager and distributed prior to the next meeting. Representatives of the following offices attend the meetings:

Dean of Administration

Registrar

Admissions

Bursar

Dean of Academic Standards

Dean of School of General Studies

Dean of Students

A smaller group started meeting in April of 1973. This paper will cover the progress of the users' group and illustrate, from the minutes of the meetings, how problems are resolved.

2. THE USERS GROUP

When meetings were first started in April of 1973, representatives from the Registrar's Office, the Dean of Students' Office and the Bursar met with the Application Systems Manager, the Director, the Project Manager and the Operations Manager from the Data Processing Department. The user offices represented at these meetings are the offices that maintain most of the data on the course and student files for the system.

It was the hope of the Data Processing Department that it could reduce its permanent representation on the committee to the project manager and encourage the other offices in the college, that would receive reports from the

file, to send representatives. To attain this goal the minutes of the meeting were distributed to all potential user offices and telephone calls pointing out particular items of importance to an office were used as a follow-up to the written communication.

Little activity occurred over the summer, but by September, two new offices, the Dean of Administration and the Dean of Academic Standards, were sending representatives to the meeting on a regular basis. Data Processing had reduced its permanent representation to the project manager who chaired the meeting.

A broad based representation of users is desirable because the reporting and evaluation needs of all offices using student and course data are not necessarily known to the offices who are responsible for maintaining the data. This lack of knowledge of the functions of other offices is not a phenomenon unique to colleges, but is equally true in industry and government agencies.

By October, 1973, the Dean of the School of General Studies Office was represented at the meetings. By the end of October, the Admissions Officer of the college, who reports to the Registrar, was also attending the meetings.

The Admissions Officer's attendance of the meetings was triggered by the minutes of a previous meeting. The next section of the paper will use the minutes of several meetings to illustrate how they have worked to bring a potential problem area to the attention of an office and how this problem was resolved.

2.1 Sharing a File. The approach we have used to the Student Information System is modular in nature. The student files are divided into three independent sub-files that are interrelatable by the system. The first of these files is the Admissions File, and the two offices that are the heaviest users of this file are the Admissions Office and the Dean of Students Office. The Admissions Office uses the files for control and reporting on admissions and applications data for

the college, and the Dean of Students uses the information for placement of students. There is some overlapping of reporting needs.

The section of the memorandum reviewing the October 17, 1973 meeting of the User Committee dated October 19, 1973 follows next. Of particular importance to the Admissions File were items 5 and 6, and distribution of this memorandum resulted in the Admissions Officer of the college attending the next committee meeting to discuss the problem of multiple applications to the college (see Item 6).

Item # 5 - Cyclical Modifications - Two aspects of the student information system are in operation. As a result of using the system, there have been requests to make modifications to some of the programs. Data Processing has made some modifications in cases where the modifications were needed to provide capabilities which were originally part of the system design. On those modifications which would provide added capabilities, however, it would be more desirable if these modifications could be made on a cyclical basis. On large computer systems, it is not uncommon to restrict modifications to a once a year or once every six months basis. This is a much more efficient use of programmers' time than regular modifications. It also allows the programming staff to concentrate in establishing new systems. This item is still open for discussion at future meeting.

Item # 6 - Students Admitted to Two Divisions of the College - A potential problem area regarding multiple admissions was discussed at this meeting. The problem specifically applies to students who take the undergraduate degree and graduate degree at Lehman College or who take an undergraduate degree and return to take other undergraduate courses in SGS. The admissions procedure requires them to file a second application and a new admissions record must be established. The student historical file must have the ability to maintain the data on one record but be able to print separate transcripts. This item will be delayed for a future meeting and in the meantime, users will discuss the problem in more depth with Data Processing. [2]

At the next meeting, on October 24, 1973, the Admissions Officer had joined the meetings for the first time. Almost the entire meeting was devoted to the Admissions File. Items 1, 2, and 3 are of particular importance. The memorandum follows.

Item # 1 - Social Security Number Verification for the Admissions File - The problem of verifying social security numbers for admissions file records was discussed at some length at this meeting. It was

stated that it is possible that the amount of error generated might not warrant any verification. It has always been the intention of the Registrar to verify this information after the student has registered and not as part of the admissions processing function. It was decided that members of the Registrar's Office, the Dean of Students Office and the Data Processing Office will look into this problem in greater detail and establish a procedure for verifying this information if the need for verification is warranted.

Item # 2 - Students Admitted to Two Divisions of the College - The problem of multiple admissions (see Item # 6 of the minutes of the meeting of October 19, 1973) was discussed at this meeting. It appears that this procedure is rather complex and a general rule does not apply to all students applying to one division and coming from another division of the college. A more in depth analysis of this problem will have to be undertaken by the Registrar's Office and the Data Processing Office. At this time, no solution has been achieved and this topic will be further discussed at future meetings.

Item # 3 - Training Session for the Admissions System - There will be a training session on Wednesday afternoon at the Fordham Center on the methods of utilizing the admissions file. It will be directed mainly to the admissions office personnel, but any administrator interested in using this module of the student system can benefit from this session. Please contact Charles Schreiber as to the exact time and meeting place.

Item # 4 - On-Line Priorities - It was mentioned that the Data Processing Office hopes to begin the design of on-line capabilities for student data within the next few months. The users should start thinking in terms of establishing which functions (registration, fee collection, grade reporting, general access, etc.) would take priority in the development of the on-line systems. Related to these functions is the question of which data elements should also be made available on an on-line basis. This topic will be further discussed at future meetings. [3]

An acceptable solution to the multiple application problem was reported on the meeting held on December 12, 1973. All offices that would be affected by the method of handling the problem had been consulted. Data Processing had proposed four alternatives. The original proposals were quite lengthy and a one line synopsis of each proposal appears below.

Proposal 1 - Adding a new data element to the file as a flag to control admissions data transference.

Proposal 2 - Manual delaying of second application processing.

Proposal 3 - Adding the necessary duplicate information to maintain the individual's record.

Proposal 4 - Adding a duplicate record to the admissions file for all duplicate applications.

The discussions at the intervening meetings had demonstrated that the users understood the implications of the alternatives proposed and were capable of evaluating the problems inherent in each. Two major factors were taken into account:

1. Ease of maintaining the data.
2. Cycle of reporting needs of several user's offices.

The minutes of December 13, 1973, Item # 1 shown below, reported on the resolution of the problem.

Item # 1 - Multiple Applications - The problem of multiple applications has been the topic of discussion at several meetings. A summary of past discussions held on this topic (see Item # 2 of the minutes of the meeting held December 7, 1973) accompanied last week's minutes. Several proposals or solutions were rendered and the general consensus is that proposal # 1 seems to be the most adequate solution to the problem at this time. Data Processing will proceed with this approach unless a better solution is rendered. All decisions on this matter rest with the Registrar who is ultimately responsible for the data. [4]

The sequence of events reported above demonstrates several things about the approach of design and implementation by committee. They are reviewed in the next section of the paper.

3. DESIGN AND IMPLEMENTATION OF A SYSTEM USING A COMMITTEE APPROACH

It is clearly demonstrated from the above sequence of events that it is feasible to get a solution to a problem through the use of a users committee approach, but it is equally clear that a participatory democracy is a slow mechanism for decision making. The question then arises why should one consider this approach. This section of the paper tries to give the advantages and dis-

advantages and the overall commitment needed so that the reader can judge for himself the appropriateness of this approach for designing and implementing a system in his environment.

3.1 Advantages and Disadvantages. The approach of a User Committee has the following advantages:

1. It offers a mechanism for reporting problems without pointing fingers at a particular office.
2. It is a fairly painless method of presenting new ideas and concepts in a workshop environment.
3. Participants feel they have helped design and implement the system and thus have a greater commitment to maintaining the system.
4. The Computer Center stops being a mysterious place that is changing the world and a threat to jobs and security.

The approach of a User Committee has the following disadvantages:

1. The decision process is slow.
2. Some solutions are compromises and not necessarily the best solution to the problem.
3. Implementation time is prolonged and some frustration develops from having discussed the capabilities a long time before they are actually implemented.

It is equally important to point out that the entire approach is only feasible if the representatives sent to the meeting from the users offices have a commitment to computerize their records and are familiar with the operational procedures in their own offices. Experience has shown us that if the manager of a department is not committed to the extent that he will free operational staff to become active participants in the project, the committee approach is a failure. This aspect of using this approach is expanded upon below.

3.2 Commitment and Approach Needed. Experience has shown us that there are certain approaches to take if success in establishing the system is to be achieved through a user group. The user group should be comprised of operational or "middle" managers. Department heads do not necessarily have the grasp of operational problems to make wise decisions concerning their computerization. This however, depends upon the size of the department. The representatives of the department must be able to make operational decisions and must be committed to the establishment of the new system. The latter condition is most important and must be shared by department heads and top administrators as well. Top management must also license and sanction the users group to make decisions. They can and should express approval or disapproval with the decisions of the users group. All policy decisions which might arise at user's group meetings should also be referred to top management. A failure on the part of top management to respond to referrals or inquiries made by the group can be interpreted as a sign of disinterest in the work of the user's group. This disinterest can spread to the members of the group, and if it does, the effectiveness of the group can be severely jeopardized. Although top management should express interest in the group, they should not be regular attendants at group meetings. An administrator or manager who is in a very influential position, can easily dominate a meeting and hinder an open atmosphere for discussion. A free and open atmosphere is most important in solving administrative problems and in establishing a new system. The chairman of the group should be aware of this and should try to guarantee that a free atmosphere does exist. This can be accomplished by using such techniques as assuring anonymity in all minutes of the meeting so that members feel that their ideas and not necessarily themselves are being recorded. Correspondence to and from the group should be addressed to the group or to the chairman and not to

individual members of the group.

4. SUMMARY

The paper reviews an approach to solving the problems of user interface inherent in using integrated data base management techniques. The approach of a user committee for design and implementation of a major system has been used successfully at the authors' college, but there are real limitations to the approach. The crucial factors to evaluate in deciding whether or not to try this approach is the length of time that is acceptable for full implementation of the system and the commitment on the part of management to computerization of their records. If it is important to have immediate implementation of a system, the committee approach is probably not a satisfactory approach to producing the interactions needed for integrated data base systems. In addition, if the managers of the groups that must interact are unwilling or unable to commit their operational people to real participation in the decisions of the committee, the approach is doomed to failure.

5. REFERENCES

- [1] Cleland, David and King, Wm.; 'Systems, Organizations, Analysis, Management: A Book of Readings'; New York, McGraw Hill 1969.
- [2] Picciano, Anthony; 'Minutes of User Meeting', October 19, 1973; Lehman College, New York.
- [3] Picciano, Anthony; 'Minutes of User Meeting', October 24, 1973; Lehman College, New York.
- [4] Picciano, Anthony; 'Minutes of User Meeting', December 13, 1973; Lehman College, New York.
- [5] Porter, R.; 'Special Problem in Applying Electronic Data Processing to Public Administration', EDP Systems in Public Management, Ed. Cornog, G; Kenny, J; Scott, E; Connelly, J; Rand McNally & Co., 1968 pgs. 91-98.
- [6] Sterdy, A.P.; 'Financial and Budgetary Problems in Large Scale Public EDP Systems', Proceedings of the Second Annual Conference on Applications of EDP Systems for State and Local Government, 1966, pgs. 118-120.

FACE TO FACE WITH DATA BASE
(ON-LINE, REAL-TIME)

Glenn B. Harvey
Director
Computer Services
DePaul University

FACE TO FACE WITH DATA BASE
(ON-LINE, REAL-TIME)

Glenn B. Harvey
Director of Computer Services
DePaul University

This paper logically follows two earlier treatments of this subject entitled GETTING TO FIRST BASE WITH DATA BASE and ROUNDING SECOND BASE WITH DATA BASE. While the over-all system in these papers is the same, the first placed emphasis on the batch maintenance aspects, the second targeted on the methods for retrieval of information, and this one describes the terminal interface that permits users to access and update their files.

INTRODUCTION - DePaul's Early Computer Environment

DePaul University has always followed a tight budget philosophy toward computers and data processing. This is altogether consistent with the University's goal of providing a cost effective program in higher education.

DePaul has approximately 10,000 students, over 525 faculty and over 40,000 graduate alumni. An additional 250,000 people have taken educational programs of some duration at the University.

The 1974-75 operating budget is approximately \$20 million, nearly four times the \$5.5 million budget of 1963-64.

Strict control on the funding of computers paid financial dividends, but left a widening instructional gap for DePaul graduates who learned little about the electronic marvel. Pressures from both internal and external sources became stronger to expand the computer resources. By late 1971, this resulted in the acquisition of new high-level staff experienced in the application of computers to both administrative and instructional systems and dedicated to the proposition that computer systems should service users in unique new ways.

DECISION TO GO DATA BASE

To lend continuity and control to Systems Planning and Development efforts, a Master Plan for University-wide Computer Services was written and presented to the Executive Committee for approval. This plan outlined the alternatives for computer services available to DePaul, their associated costs and benefits.

Included were recommendations for hardware, software, staff and an implementation schedule for application systems.. Objectives to be achieved by new systems required that they be:

- * Responsive - To changes in user needs regardless of timing of source of change.
- * Flexible - Able to provide for new files, new data in old files, new methods of processing old data, entire new systems, or whatever need arises.
- * Simple - Easily understood by users of data, as well as the computer technician.
- * Self-documenting - System should provide relatively complete documentation of itself as a by-product of its own operation.
- * Parameter-driven - Most functions to be handled by entry of control cards or tables with tailored programming held to a bare minimum.
- * User-controlled - User should have high degree of control over his own data files, their content, maintenance and usage.
- * Easily-converted - Little difficulty converting existing 1401 files from old to new systems.
- * Economical - All of the above goals should be possible on relatively low-cost computer equipment with small memory and few input/output units. Staff expenditures should likewise be minimal yet able to accommodate system demands and changes.

Hardware selected was an interim System 360/30, 96K memory, 4 tape/3 disk, computer to be followed by an IBM System 370/135, with the same memory and input/output units, when available in January 1973. The 3 disks were originally 2319, replaced by double-density ITEL disks in August, 1972, thus making approximately 175 million bytes of storage available on-line. The interim hardware was installed March, 1972.

It was recommended and approved that the new machine system be constructed using Data Base concepts. That is, files within the "data base" would be logically inter-related in a network fashion. This had the obvious advantage of permitting the development of application systems that could easily retrieve new "relationships" of data, i.e., new information for new uses, rather than just the data themselves. Traditional file-oriented machine systems are programmed to contain specific data relationships that satisfy needs at a point in time, but are time consuming and costly to reprogram when changes occur in the needs for information.

SELECTION OF DATA BASE TOOLS

Several decisions were also made in regard to existing computer software that could aid DePaul's development of a Data Base System. Much time was devoted to seeking out and evaluating "packages" that purported to cure an organization's problems.

While investigating one such product (a data base management system) called TOTAL, we learned of its use at Amherst College in Amherst, Massachusetts. Amherst had developed a generalized maintenance system using TOTAL as the "super-access" method for organizing its disk files. Their approach to data processing problems appeared to us more advanced, responsive, and flexible than any other we found. The Amherst example became the foundation for the DePaul system herein described.

For our entry into the world of Teleprocessing, we selected a package called CRT-Interface, marketed by Westinghouse Tele-Computer Systems of Pittsburgh. This package requires minimal memory for operation (32K partition), permits COBOL programming of TP jobs and cost DePaul \$900.

DATA BASE SYSTEM STANDARDS

The DePaul System is really a Data Management System. It is composed of four basic parts:

- *Generalized Maintenance System (GMS)
- *Generalized Retrieval (and Reporting) System (GRS)
- *Generalized Utility System (GUTS)

The fourth part permits TP inquiry/update into the Data Base and is called Generalized Terminal System (GTS). This paper principally treats only the last system, our terminal interface to the user.

GMS Job Stream

A single batch job stream GMJ001 exists for the maintenance of any and all files in the Data Base. All files can be active in the same run, that is, incoming transactions can be targeted for any Data Base file. It is expected that only one GMS run will be required each day to maintain all such files. The System Flow in figure 1 shows input transactions being converted to disk via the IBM card-to-disk utility. Program GM0020 edits each transaction thoroughly, flags those with errors and releases them all to the update program GM0030. GM0030 is a calling program that calls the generalized module XX0900 to process all direct changes to the appropriate Data Base files:

A direct change transaction uses one of the transaction codes shown in Figure 2. In effect, such transactions affect no file or item of data except that which is specified in the transaction itself. An example would be the recording of a student being registered in a specific course.

GM0030 can also call numerous other update modules for the purpose of making indirect changes. Tailored programs, such modules are triggered by the type of transaction entered into the job stream. Thus, in any given run, many different types of files, activity, and indirect changes can be involved. An example of an indirect change could be the setting of an Enrollment Status Indicator, in a different file, when a student enrolls for any course.

GM0040 prints all error, unposted transactions that entered a given run, while GM0050 prints the posted conditions. In addition, GM0050 analyzes the users of data posted and generates, to multiple users if necessary, monitor reports of records that have changed:

Parameter-driven GRS

At any time, and at regularly scheduled reports, the GRS capability permits extraction of records from one or more Data Base files with or without sorting, for the printing of reports (lists, labels, accounting reports, statistics, etc.) all by parameter entry. At least 80% of the computer center's output is handled by the generalized Retrieval (& Reporting) System.

Data Base Standards of Construction

Constraints on construction of the DePaul Data Base were established in order to use the GRS to maintain all files with but one update module. First, some definitions are in order.

*Physical File = Grouping of similar records within the Data Base. TOTAL permits two types, master or single-entry and variable or variable-entry. Each is identified to TOTAL by a four-letter name, such as PEOP (for a PEOPLE file of names, etc.)

*File Record = Logically unique data grouping for a given primary control (like Soc. Sec. #) in the case of a master file. A variable file record needs both a primary control segment and a secondary control (part of SEGI) to establish its uniqueness--and thus GRS can find the record when trying to update it.

*Segment = Grouping of data elements within a file record. CTRL segment contains only the primary control element.

A data segment, i.e. SEGI, can contain any number of data elements.

*Data Elements = Usually the smallest data component of a segment.
System permits multiple definition of elements

While the GMS programs could be modified to accommodate almost any type of Data Base system, the setting of standards within our own has greatly facilitated the development of new methods and techniques for dealing with the data contained therein.

Data Base Dictionary of Elements

Several files in the Data Base are used solely for control of the system. A schematic of them is shown in Figure 2A. Perhaps the most important is the DICT or Dictionary of Data Element File. Every element is described in DICT, its associated file, segment, beginning position, size, class, etc. Each element has a unique number and is structured in a special way as seen in Figure 2. An element in the DePaul system is related to physical file by its prefix.

In the final analysis, the DICT even describes itself, as seen in Figure 4. Here, it can be seen that 22 elements comprise one DICT record, 21 of which are in SEGI and 1 in CTRL segment. A 23rd element really exists in DICTSEGI positions 1 - 6 known as "last transaction date" which is automatically supplied and maintained by the GMS. Because the DICT can describe itself, the GMS can be used to update the DICT records of any element. In other words, changes to the structure of the system are as simple as changes to the data.

Data Base File Relationships

Another system control file FILS describes the characteristics of each physical file and its relationship to others in the Data Base; see Figure 3 for a sample Data Base Schematic and Figure 4 for the associated content of the FILS file. An interesting fact is that FILS can exist under any data base access method (not necessarily TOTAL). File relationships are also carried in the TOTAL core-resident descriptor module, but that is frequently proprietary; changeable or the vendor may even be changed. Like other data-base files, FILS can be modified through GMS. FILS and DICT along with the attendant standards of data base construct, terminology and file relationships, form the heart of the DePaul system.

Data Base Input Processing

Perhaps one of the most interesting parts, and certainly the watch-dog of this system is Input Editing. This is also generalized and is a separate front-end sub-system. Its purpose is to provide broad capabilities for entering either single-element or multiple-element transactions in a relatively free-form, highly flexible manner that insures that transaction values are correct.

System Control Files involved include the FORM master file and FUSE (Form USagE) variable file that together define the detail layout of multiple-element transactions. For example, in Figure 4A, assume a certain input form #F35 requires that two cards be punched with data comprising three elements on the first card and ten elements on the second. These files would show the following:

*FORM - designates the Form #F35, max. no. of cards = two; Primary Control # (Soc. Sec. #) starts in cc 1 and ends in cc 9, (element data thus starts in cc 10) and a pointer to a FUSE file chain of records where:

*FUSE - For card #1, Form F35, first element value in cc 10-40, second element value in cc 41-50 and third element in cc 51-68. The appropriate element no. for each is also given.

For card #2, ditto for 10 elements.

With this information, the system can "explode" multiple-element transactions into their component single-element versions that are acceptable to the GMS update modules.

When a single-element transaction is identified (or created via "explosion"), the DICT record for that element no. is read from the Data Base. Armed with its element characteristics, validity of the transactions data class, length of value being entered, authorization of dept. no., validity of "system" no., etc. can be determined.

Then, two new System Control Files TABL and VALU are brought into play. (Another master and variable file combination, like FORM/FUSE, these files contain the range of expected, or precise, values for any given element. The transaction value is subjected to the proper validity checking by these files before successfully passing into GM0030.

Some Hardware Considerations

The batch or TP system is designed to occupy no more than 48K of memory for any program module. For instance, the update phase GM0030 can include in memory at one time:

- *Root Module GM0030, Macros
- *TOTAL Data Base Description Module (Size varies)
- *TOTAL Data Base Manager
- *Transient COBOL modules like XX0900

A 96K IBM 370/135 easily accomodates this system at a very modest hardware cost. Programming language for all primary modules is IBM COBOL D. There are also 3 macros written in Assembler, each about 15 instructions. The system is thus highly hardware and software-independent. Operating system is DOS.

Three additional installations of the batch system have been made, two of which are OS with programs converted to ANS COBOL.

Our double-density ITEL disk drives are able to provide concurrent operations among (1) 1401 Emulation, (2) Data Base Batch Processing, (3) Data Base TeleProcessing, and (4) Spooling of I/O operations.

DATA BASE FACE TO FACE - GENERALIZED TERMINAL SYSTEM

System Components

A rather intricate combination of hardware facilities and special software is needed to build a teleprocessing system. The following discussions of the subject are purposely simplified and, hopefully, clarify DePaul's Generalized Terminal System.

1. Hardware Employed

- * Configuration - IBM 370/135 configuration, including the TP terminal network.

- * Terminals - Five CRT terminals of the IBM 2260 type are used. Screen capacity is 960 characters spread over 12 horizontal lines of 80 characters each. The terminal keyboard layout is much like a typewriter with the addition of several special-function keys.

Pressing the Start-of-Message (SOM) key creates a character symbol ► that is the beginning of the message to be sent to the computer. The TRANSMIT key activates the computer to read the screen's current message and places an End-of-Message (EOM) ◻ symbol in the cursor position.

Thus, the characters ► ABCD◻FGH would result in the computer reading only the ABCD.

*Network Operation - One control unit services all 5 terminals by creating a polling sequence to repeatedly check each terminal for activity. The computer actually communicates with the control unit, only, by issuing read-screen or write-screen commands.

Such commands state the specific terminal number involved and the expected length of the read/write character string, up to 960 characters.

Writing, from computer memory to terminal screen, always starts at the first character position on the screen, at the upper left or 'home' position. The operator's response to the last screen written is what is read from the screen to computer.

The message being read is always defined by the hardware as the characters on the screen that lie between the SOM and EOM characters symbols. Only one SOM or EOM can be on the screen at one time - this is generally enforced by the hardware.

We have a local terminal network, where in no 'remote' or phone transmission is involved. Terminals may be placed up to about 1000 feet from the control unit connected by a cable.

Local networks are less expensive than remote since no data transmission facilities are needed, and have extremely fast screens (read or write time) . . . virtually instantaneous.

However, the time to read or write a screen is not a significant part of total 'response time' as commonly defined. Response time is considered the time it takes for the computer to 'answer', that is, to begin writing a screen in response to something the operator entered. Depending on the circumstances, response time varies greatly from one teleprocessing system to another. The DePaul system is quite good in this respect, averaging about 3 seconds.

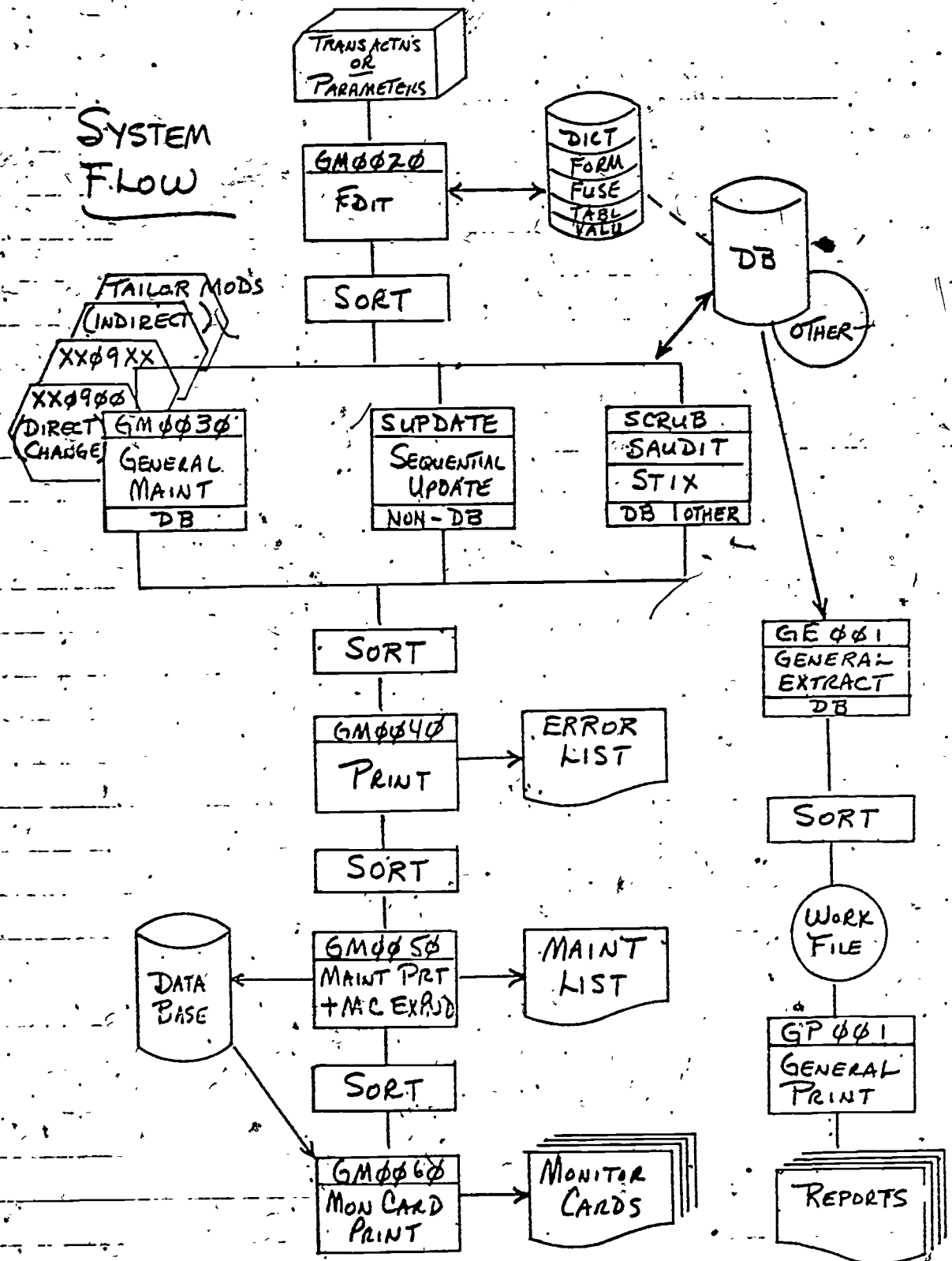


FIGURE 1

STANDARD DATA BASE TERMINOLOGY

- System No. = AA 2-digit alpha
- System Logical File = Usually 1 mast file with some linked Var Files
- Such as RG = Registrar
AD = Admissions
- File No. = AA (Standard Abbreviation) or 8AAA (Total Name)
- Physical File = Master or Variable
- Such as PE = PEOP
CR = COUR
- Data Element No. = AA (File Abbreviation) or 8AAA (Element #)
- Identified by Physical File
- AA331 required for Master or Variable Files and Control # (up to 30)
- Such as PE331 = PEOP SS#
PE333 = PEOP Last Name
- Can be redefined in DICT to designate combinations of other elements or the unique processing of the given element.

STANDARDS FOR DATA BASE CONSTRUCTION

- Each Physical File Record =
 - Control Segment (up to 30 characters)
 - Data Segment(s) (up to 240 characters)
 - Master File can have 1 to 9 segments
 - Variable File can have 1 segment
 - Segment Name is "SEGN" (e.g. SEGP)
- Master File
 - 1 of links to Variable Files = 1 to 20
 - First 6 digits SEGI = Last Trans Date
- Variable File
 - All updating done via a single Base Master-File
 - Must have Record Code (for TOTAL)
 - Has Record Type first 2 digits SEGI
 - Can have additional secondary control following Record Type

TRANSACTION CODES

- AA = Add Master Record
- DD = Delete Master Record and any associated Variable Files
- CR = Change element in Master or Variable Record by replacement
 - or-
 - Create new Variable Record if none exists and insert element value from this transaction.
- CE = Change element in Master or Variable Record by Ensure (set to blanks or zeroes as appropriate)
 - and-
 - Completely eliminate a Variable Record if this was its last data element remaining.
- CA = Change element by numeric addition or subtraction.

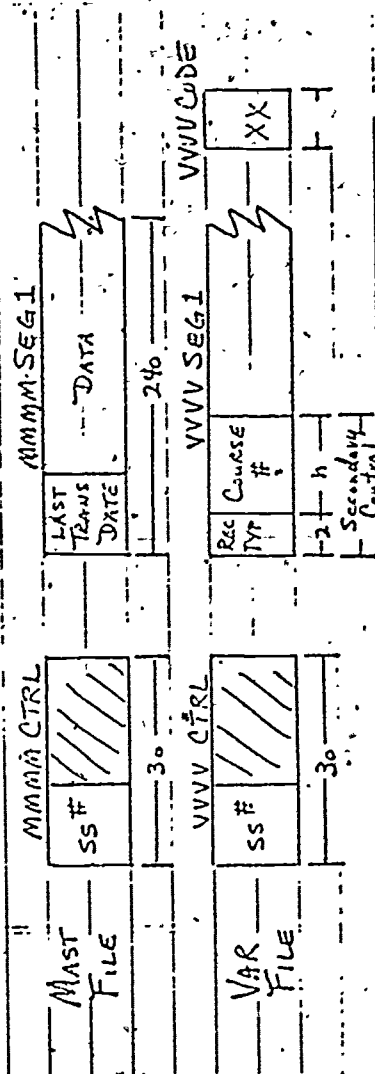


FIGURE 2

DEPAUL SYSTEM CONTROL FILES

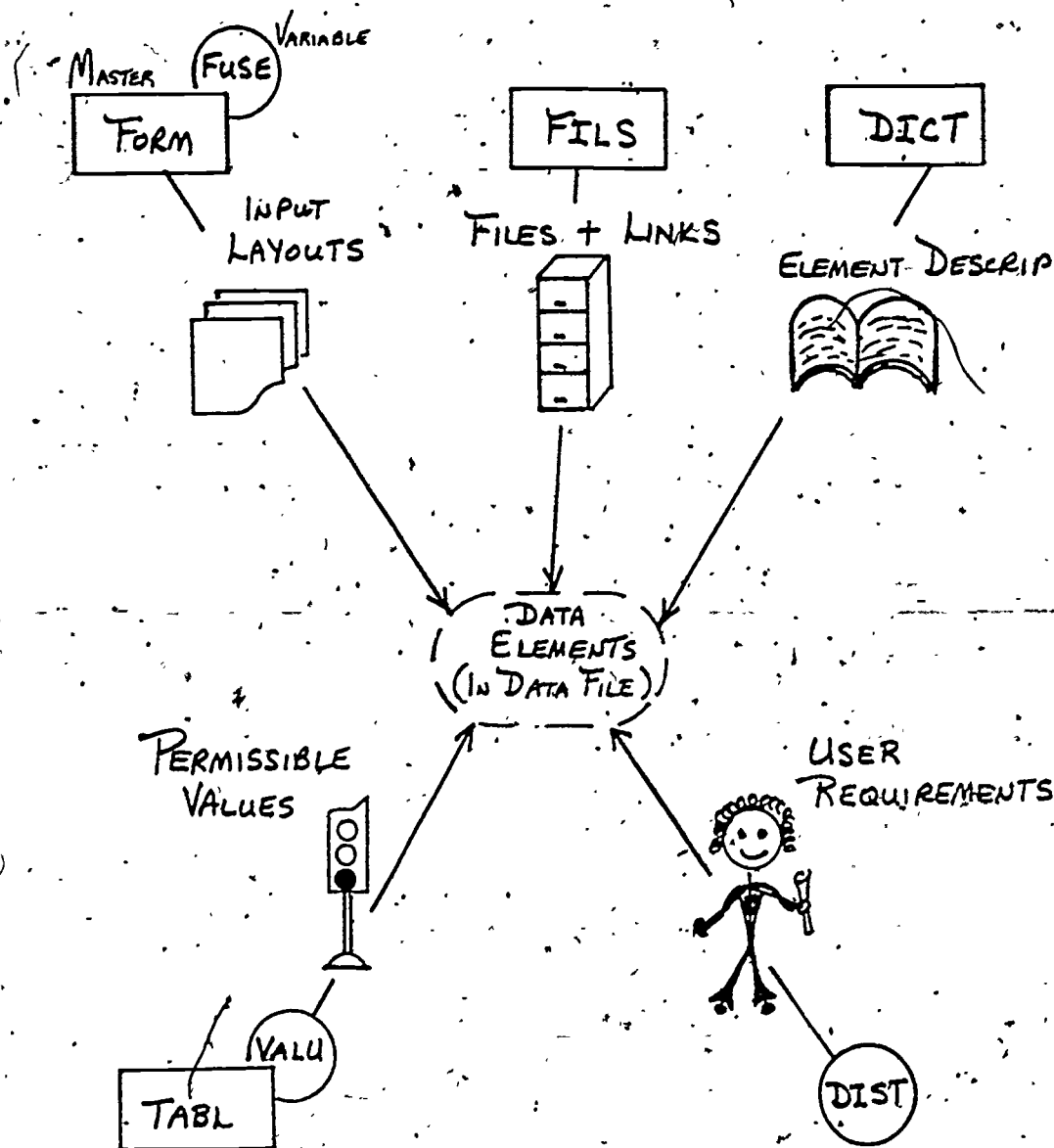
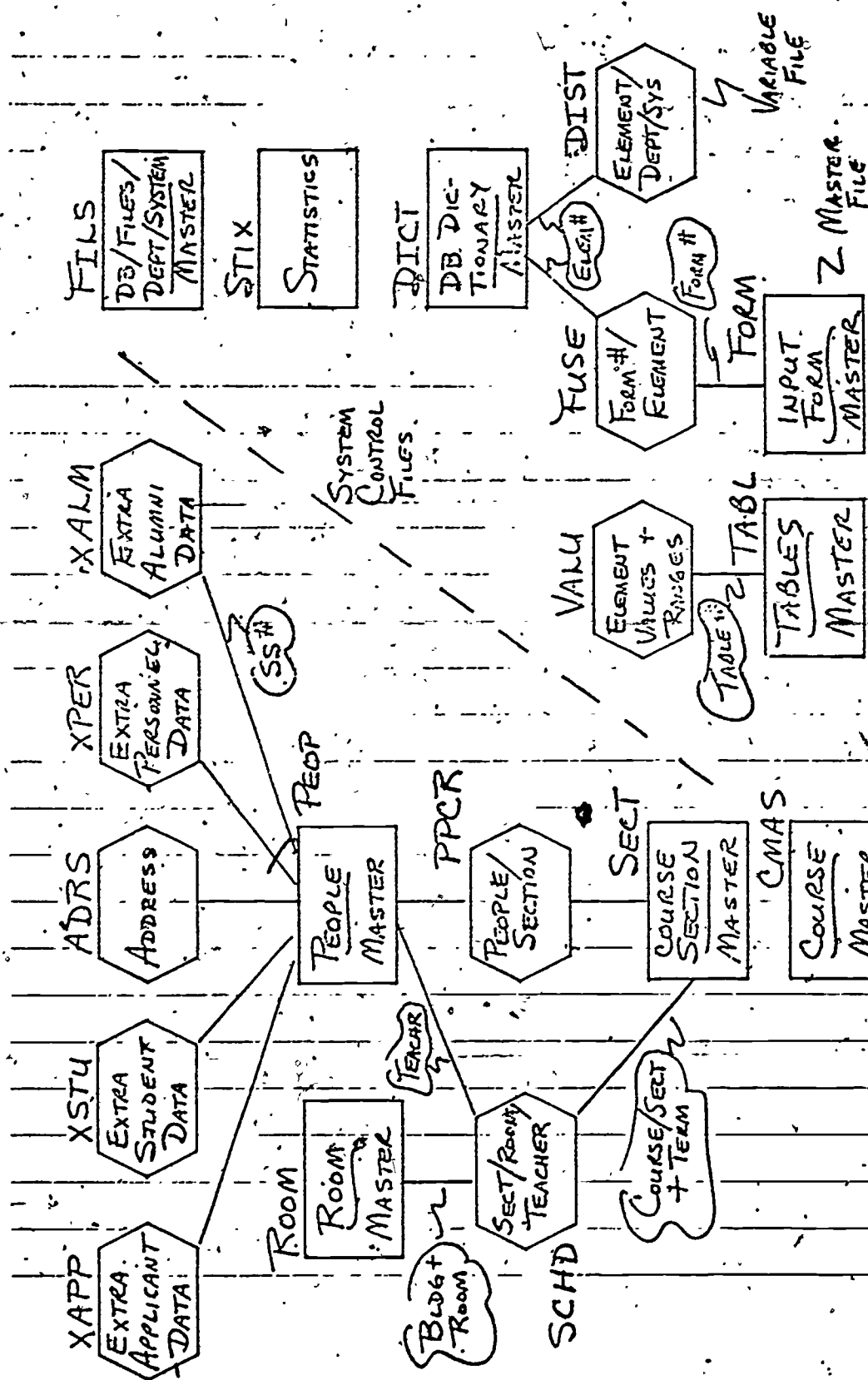


FIGURE 2A



DEPAUL DATA BASE

FIGURE 3

INITIAL LOAD OF DATA BASE FILES DATA ELEMENTS 09/01/72

Vol. II

284

NO.	M/V	SGS	LKS	UPDT	LY	LK01	LK02ETC.
FILE	M	1	00					
DICT	M	1	00					
PFOP	M	2	07			0	PPCR	ADRS1
COUR	M	1	01			0	PPCR	
PPCR	V	1	0	PEOP	01			
ADRS	V	1	0	PEOP	02			
MF01		0	04			0	FILE	DICT PEOP COUR
VF01		0	02			0	PPCR	ADRS
PE0114	B	2	007020			027003		
PC0114	B	2	007020			027003		
AC0114	B	2	007020			027003		
PE0115	A	3	007012			030004	027003	
CR0114	A	1	007015					

INITIAL LOAD OF DATA BASE DICTIONARY DATA ELEMENTS 09/01/72

NO.	M/V	RCOD	FILE/SEG	SIZE	LOC	CLS	DEC	CHG	NAME	DESCRIPTION	DEPT	R	SHIFT	UPD	TBL	RTYP
D1001	M		DICTCTRL	5	1	X			D1001	DATA ELEMENT NUMBER	114	C	000			
D1020	M		DICTSEG1	1	7	X	ER		D1020	FILE TYPE	114		000			
D1030	M		DICTSEG1	2	8	X	ER		D1030	RECORD CODE - TOTAL	114		000			
D1040	M		DICTSEG1	8	10	X	ER		D1040	SEGMENT NAME / FILE NAME	114		000			
D1050	M		DICTSEG1	3	18	N	0	ERA	D1050	ELEMENT LENGTH 2BYTES	114		000			
D1060	M		DICTSEG1	3	21	N	0	ERA	D1060	ELEMENT LOCATION 3LEFT	114		000			
D1070	M		DICTSEG1	1	24	X	ER		D1070	ELEMENT CLASS	114		000			
D1080	M		DICTSEG1	1	25	N	0	ERA	D1080	DECIMAL PRECISION	114		000			
D1090	M		DICTSEG1	3	26	X	ER		D1090	VALID CHANGE TYPES	114		000			
D1100	M		DICTSEG1	6	29	X	ER		D1100	ELEMENT MNEMONIC	114		000			
D1110	M		DICTSEG1	25	35	X	ER		D1110	ELEMENT DESCRIPTION	114		000			
D1120	M		DICTSEG1	3	60	N	0	ERA	D1120	MAINTENANCE DEPARTMENT	114		000			
D1130	M		DICTSEG1	1	63	X	ER		D1130	CHANGE RESTRICTION CODE	114		000			
D1140	M		DICTSEG1	3	64	N	0	ERA	D1140	SHIFT LOC 2AUTOM	114		000			
D1150	M		DICTSEG1	3	67	N	0	ERA	D1150	UPDATE ROUTINE	114		000			
D1160	M		DICTSEG1	2	70	X	ER		D1160	VALIDATION TABLE NO.	114		000			
D1170	M		DICTSEG1	2	72	X	ER		D1170	RECORD TYPE - CODED REC.	114		000			
D1180	M		DICTSEG1	3	74	N	0	ERA	D1180	SECONDARY CONTROL LOC. LF	114		000			
D1190	M		DICTSEG1	2	77	N	0	ERA	D1190	SECONDARY CONTROL LENGTH	114		000			
D1200	M		DICTSEG1	1	79	X	ER		D1200	MULTIPLE OCCURENCE CODE	114		000			
D1210	M		DICTSEG1	1	80	X	ER		D1210	ESSENTIAL/NON-ESSENTIAL	114		000			
D1220	M		DICTSEG1	1	81	X	ER		D1220	MONITOR CARD CODE	114		000			

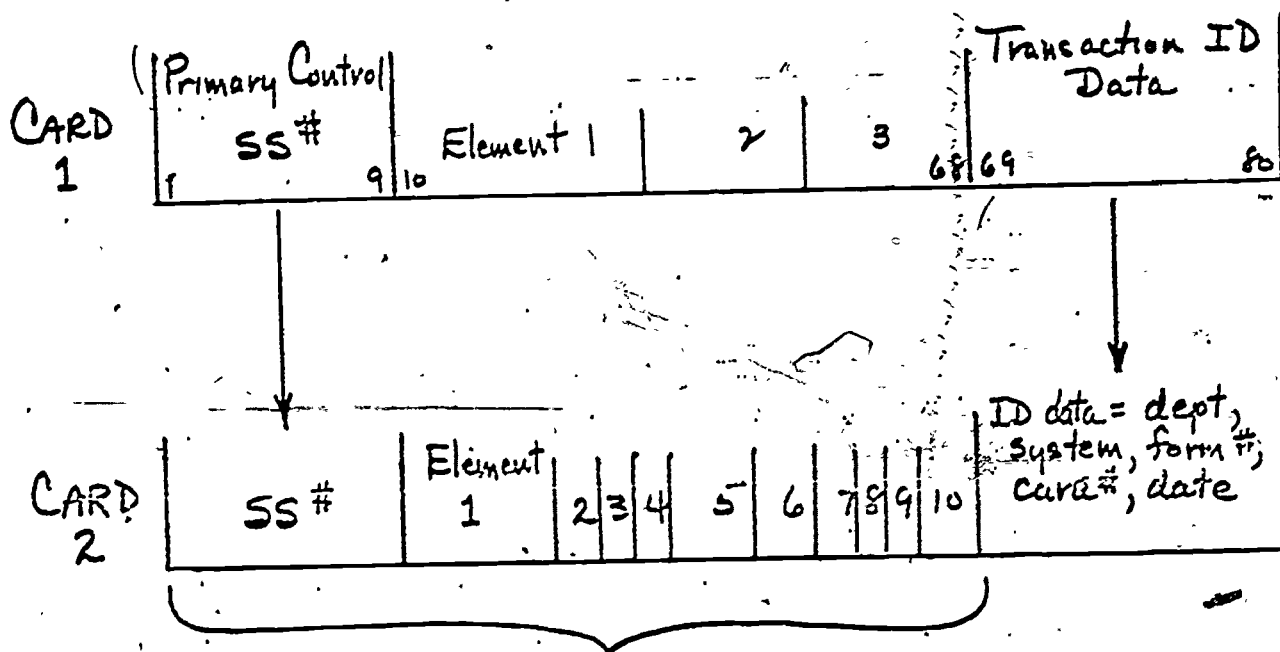
INITIAL LOAD OF DATA BASE DICTIONARY DATA ELEMENTS 09/01/72

NO.	M/V	RCOD	FILE/SEG	SIZE	LOC	CLS	DEC	CHG	NAME	DESCRIPTION	DEPT	R	SHIFT	UPD	TBL	RTYP
PE001	M		PEOPCTRL	9	1	N	0		PE001	PEOPLE FILE CONTROL NO.	114	C	000	900		
PE010	M		PEOPSEG1	12	7	X	ER		PE010	LAST NAME	114		000	900		
PE020	M		PEOPSEG1	8	19	X	ER		PE020	FIRST NAME	114		000	900		
PE030	M		PEOPSEG1	3	27	X	ER		PE030	CLASS	114		000	900		
PE040	M		PEOPSEG1	4	30	X	ER		PE040	MAJOR	115		000	900		
PE050	M		PEOPSEG2	1	9	N	0	ERA	PE050	ALUMNI STATUS	114		000	900		
PE060	M		PEOPSEG2	1	10	N	0	ERA	PE060	ADMISSION STATUS	114		000	900		
PE070	M		PEOPSEG2	1	11	N	0	ERA	PE070	ACTIVE STUDENT STATUS	114		012	900		
PE080	M		PEOPSEG2	1	12	N	0	ERA	PE080	STUDENT STATUS - PREV	114		000	900		
PE090	M		PEOPSEG2	3	15	X	ER		PE090	GEORGIAN CODE	114		000	900		
PE100	M		PEOPSEG2	10	16	X	ER		PE100	SUBJECT ADDR - LOCAL	114		000	900		
PE110	M		PEOPSEG2	8	28	X	ER		PE110	CITY - LOCAL	114		000	900		
PE120	M		PEOPSEG2	2	34	X	ER		PE120	STATE - LOCAL	114		000	900		
PE130	M		PEOPSEG2	5	36	X	ER		PE130	ZIP CODE - LOCAL	114		000	900		

FIGURE 4

GENERALIZED EDITING

INPUT FORM # F35



EXPLODED INTO SINGLE
ELEMENT TRANSACTIONS

FIGURE 4A

CONCURRENT ADMISSIONS IN AN
UPPER LEVEL UNIVERSITY

Donald L. Linebarger
Systems Analyst
Administrative Data Processing
University of Texas at Dallas

1. INTRODUCTION

The University of Texas at Dallas (UTD) is an upper-level university located in Richardson, Texas, which is a northern suburb immediately adjacent to Dallas. It was originally a private research center. In 1969, it became a part of The University of Texas System. UTD immediately began offering doctoral level work, and in the fall of 1972, initiated several master level programs to support the doctoral programs. In the fall of 1975, The University of Texas at Dallas will become an upper-level university by enrolling its first undergraduate students at the junior and senior levels.

UTD is a user of the University of Texas Regional Computer Center in Dallas, Texas; which also supports The University of Texas Health Sciences Center in Dallas, The University of Texas at Arlington, and other smaller users. The present hardware is an IBM 370/155, five tape drives, four 3330 disk drives, and other peripheral equipment. The present software is OS/MVT, TOTAL Data Base Management System; and no TP Monitor or Query Language.

A standing general requirement of the administration of The University of Texas at Dallas is to receive high quality information with a maximum turn-around time of one day. This implies some kind of on-line inquiry and updating capabilities. It implies file structuring for efficient accessing of data elements. It implies a query language with supporting software to be able to quickly describe inputs, outputs, and processing to be done. With this in mind our approach was to make any provisions we could anticipating a TP Monitor and query language at a later date, to

develop batch systems which access records in direct or sequential mode using the TOTAL Data Base Management System, and design all data collection forms and reports which will be replaced with on-line inquiries to fit within the physical limitations of a CRT screen.

With a very small research-oriented faculty and student body, the University needed to begin a program to create interest in UTD among prospective enrollees for the fall of 1975. The program chosen is called Concurrent Admissions. It is designed much like a regular admissions system, but is flavored heavily with counseling services and does not require the prospective student to supply more than what is asked on the application. The program provides advanced counseling to prospective students of the University while they are completing their first two years. The bulk of the students participating are enrolled in area community colleges, and are in a position to visit the UTD campus for counseling as often as they feel necessary. However, UTD counselors also travel extensively to community college and high school campuses to reach as many prospective students as possible. To support the effort, a computer system was designed and is, at this time, approximately ninety-five percent implemented.

The Concurrent Admissions System is situated logically, as in Figure 1, between a variety of documents from and to prospective students, and the regular undergraduate admissions system. It provides advanced counseling to prospective students of UTD, while they are completing their first two years at a community college or some other four-year college. The system is promotional in nature and is where the student's record may be created initially; and be passed through many other systems until the student graduates or leaves UTD.

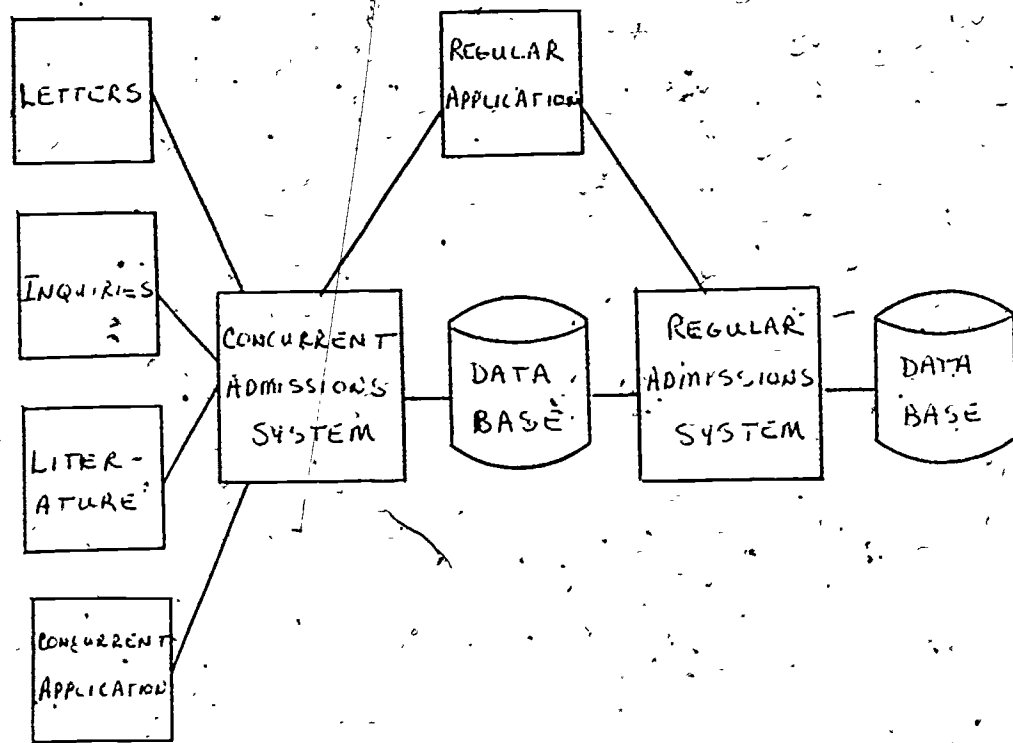


FIGURE 1. Concurrent Admission System Environment

The Concurrent Admission System itself can be thought of as five general areas. They are: (1) prompting inquiries; (2) prompting Concurrent Admissions applications; (3) creating and maintaining a Concurrent Admissions data base; (4) using the data base; and, (5) prompting regular undergraduate applications for admission.

1.1 Prompting Inquiries. Inquiries about undergraduate degree programs are prompted as shown in Figure 2 by placing newspaper ads, radio and TV spots, having booths in local shopping centers, displaying posters at advantageous locations, and by sending letters to all people on such lists as: (1) National Merit Scholars; (2) Dallas/Fort Worth High School Graduates; and, (3) Educational Testing Service's Minority Search, Valedictorians, and Salutatorians. Additional inquiries result from (1) personal letters, (2) walk-ins, and (3) phone calls. A standard Inquiry card is filled out in the office as each inquiry occurs. Another source of "inquiry" is from ACT and SAT Test Score Tapes. Each entry on these tapes is treated as an inquiry and entered in the data base,

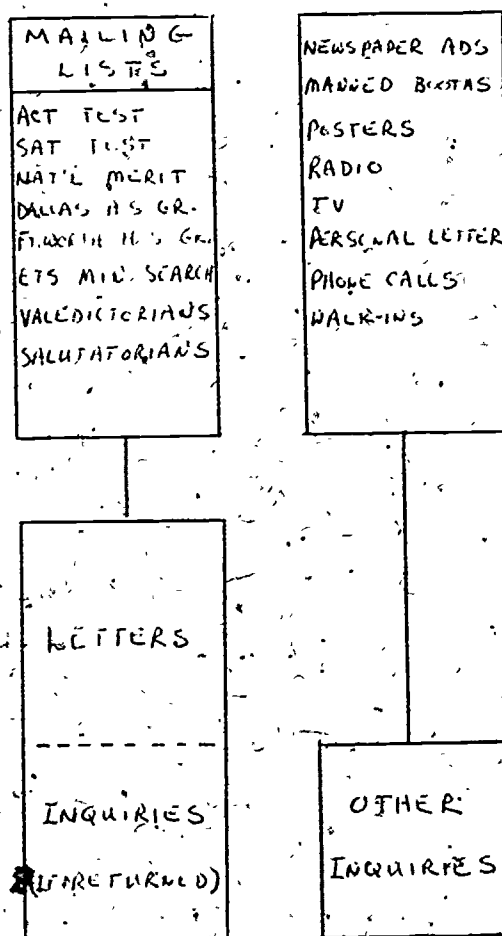


FIGURE 2. Prompting Inquiries

1.2 Prompting Applications. Applications (Appendix 1) for Concurrent Admissions are prompted, as shown in Figure 3, by (1) counselors traveling to area campuses and talking to prospective UTD students, and, (2) periodically having the computer examine the data base and generate a follow-up letter to be sent along with a blank application to all 'inquiries' who have not yet submitted an application.

To supplement these direct efforts, pertinent literature is sent to selected groups indicating a certain major or attending a certain institution, etc.

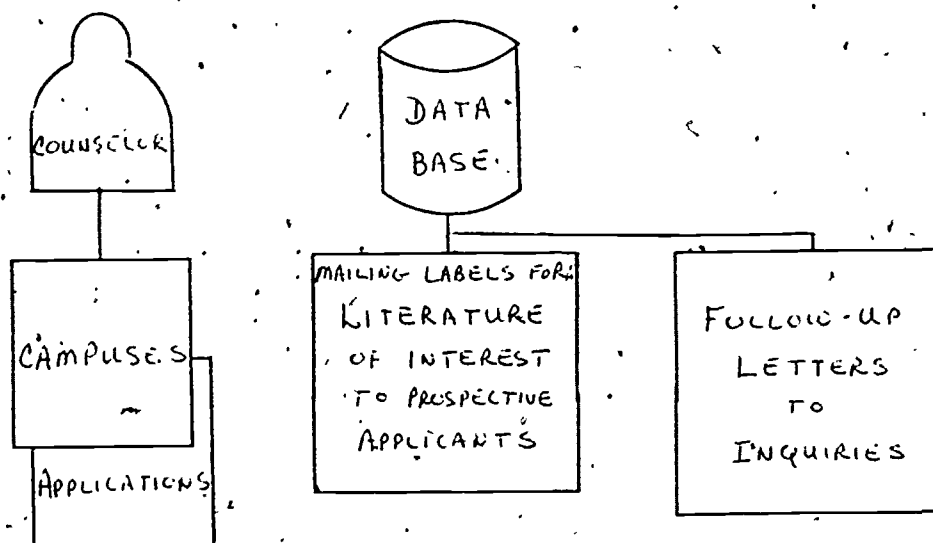


FIGURE 3. Prompting Applications

1.3 Maintaining The Data Base. The data base is maintained as shown in Figure 4 by adding all applications and inquiries to the file on a weekly basis. They are edited. The data base is updated. Mailing labels are created for first time inquiries. Back-up and recovery procedures are executed.

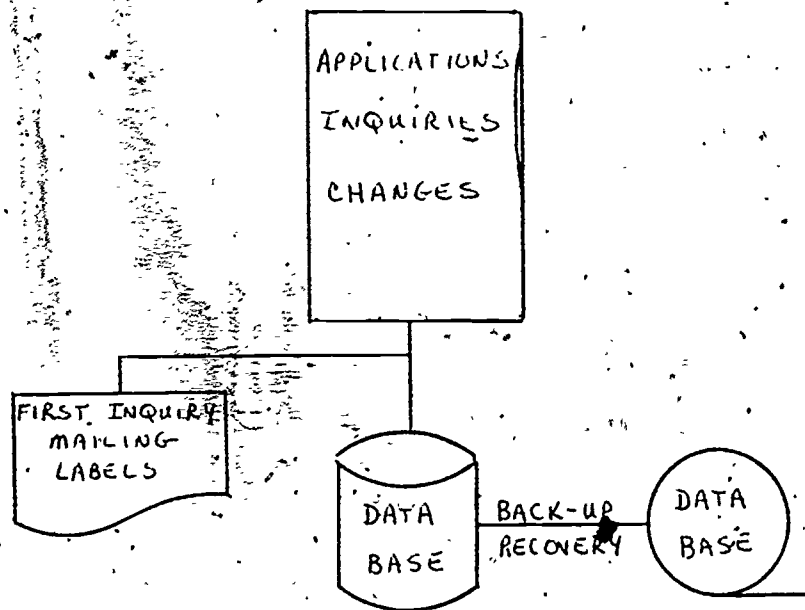


FIGURE 4. Create and Maintain Concurrent Admissions Data Base

1.4 Using The Data Base. The data base is then used for a variety of purposes as shown in Figure 5. They are: (1) evaluating applicants for admission; (2) re-evaluating rejected students for admission assuming possible submission of more current data; (3) doing various lists and summaries; and, (4) making mailing labels. A use of major interest is that of evaluating students for acceptance. They are accepted in one of four ways: (1) rank in the top half of his high school graduating class and make at least 18 on the ACT, or 800 on the SAT; (2) complete at least 27 semester credit hours with a grade point average of at least 2.5 on a 4.0 system; (3) complete at least 54 semester credit hours with a 2.0 average on a 4.0 system; or, (4) be accepted on individual approval. Those accepted are sent an acceptance letter and a View Book about UTD. Those rejected are sent a rejection letter explaining conditions he/she must

meet to become accepted. Another important use is providing lists for counseling purposes. Still another very valuable use is for planning facilities and Student Services.

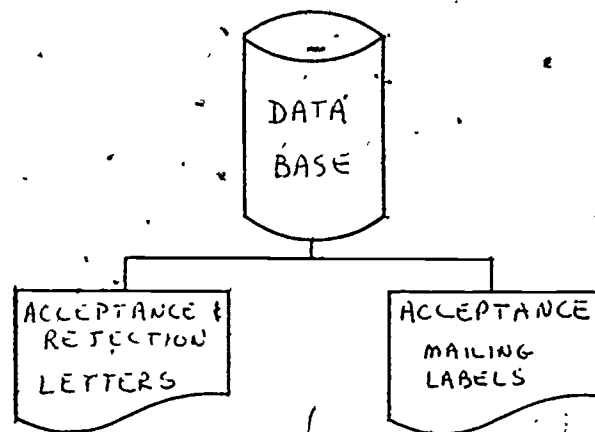


FIGURE 5. Use Data Base

1.5 Prompting Regular Undergraduate Applications. The last general area in the system is, as in Figure 6, prompting the student to apply for regular admission. The expected date of enrollment is examined, and for all students indicating first enrollment for the coming semester, a computer generated letter is written explaining the steps for regular admission. There is one letter for those who have been accepted and another for those who have not. The letter to the unaccepted ones explains any deficiencies and reiterates the requirements.

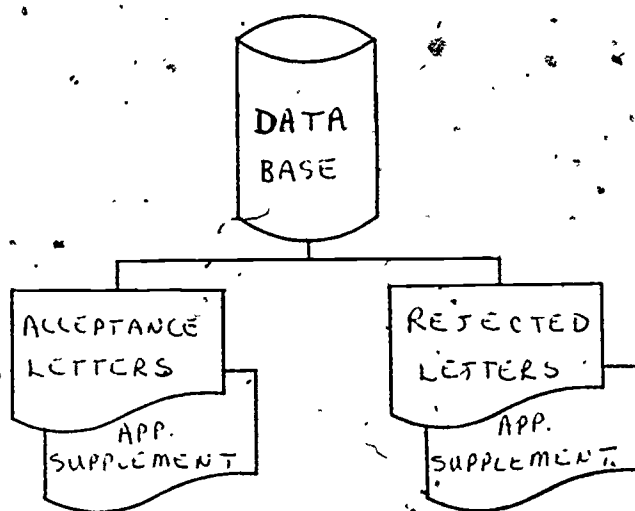


FIGURE 6. Prompting Applications for Regular Admission

2. THE DATA BASE

2.1. Data Elements. The data elements needed in the system were decided upon at meetings of a group of administrators designated by the Academic Council. The participants were people such as the Director of Admissions, Registrar, Director of Student Services, the Assistant to the Vice President for Academic Affairs, and others. As each expressed his or her information needs about students scheduled to begin enrolling in approximately two years, a list of data elements emerged and an application for Concurrent Admissions was designed. One document supplies all information needs expressed by the aforementioned parties as in (Appendix 1).

2.2 Logical Data Groups. The next step was to decide where each data element should be maintained. In TOTAL, there are two kinds of files meaning that for any key there is one and only one record in the master file. Master file records are read directly, and point to the first record(s) of all chain(s) linked to this master record. Variable files are multiple entry files meaning that the number of records in a variable file is

limited only by the physical limits specified in the Data Base Generation (DBGEN). Variable files must be attached to a master file as the address of the first record of a string is maintained only in its master record. Each record in a string is linked both forward and backward, allowing application programs to 'write before' or 'write after' a variable which has already been read.

The decisions made here will have far-reaching and long-range effects on the efficiency, effectiveness, flexibility, and maintainability of the system. The designer must seek answers to such questions as: (1) Will disk space be a constraint? (2) How, and how often, is each data element used? (3) Does each element exist only once for a primary key? (4) Does it always exist? (5) Can each element exist more than once for a primary key? (6) Which set of elements are used for each logical system function? (7) Should all singly occurring elements be in the master record? (8) Should they be in the master if they don't always exist? (9) Should all elements be in variable records? (10) Should they all be in one variable record? (11) Should they be grouped in variable records according to logical system functions? (12) Should they be grouped in variable records according to frequency of usage? (13) Which arrangement of elements allows the simplest application program logic? (14) Which arrangement of elements allows the most efficient execution by application programs? (15) Which arrangement most easily allows expansion of the data base? (16) etc. The answers to these questions must be determined within the framework of the functions the system

must perform, and the resources available to develop and support the system. Our decision was to minimize resources. This then lead to the development of the Data Base File Chart.

2.3 Data Base File Relationships. In minimizing resources, we had to:

- (1) simplify program writing and maintenance;
- (2) use as little permanent disk space as possible;
- (3) provide for easy back-up and recovery;
- (4) provide for easy expansion of the data base;
- (5) minimize execution time; and
- (6) start simply enough as not to require excessive training and planning before beginning.

Since the system is extremely person oriented, we agreed that the only type of on-line inquiry and updating we would have to have later would be by person. All other summaries and reports could be run on a remote batch printer. This then made the File Chart a simple one. The master file contains all data elements occurring only once.

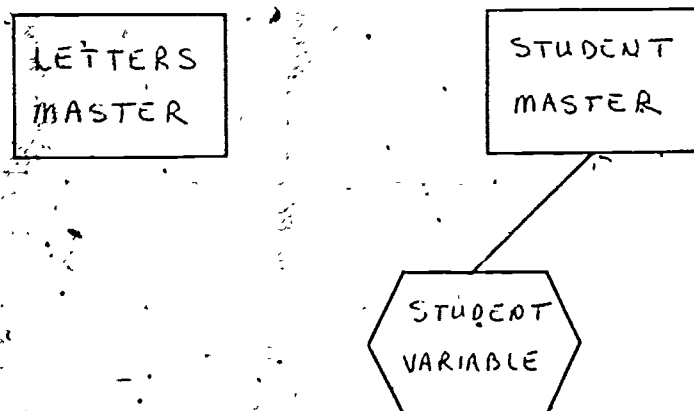


FIGURE 7 Data Base File Chart

The variable file contains all data elements occurring more than once.

In complying with our criteria for minimizing resources, we now had to determine how each logical system function would be accomplished with this data base. The system functions are: (1) update student records; (2) evaluate student records for admission; (3) write letters to students; and, (4) inquire into the data base; on-line by student (later), and other summaries and reports (now, batch).

Program logic is simplified because updating, evaluation, letter writing, and on-line inquiries are all done by student; and all data for a student is in the master record except the correspondence log which is kept in the variable file and is a by-product of the evaluation. The other summaries and reports are done by extracting the master file and writing a batch summary and/or report.

Disk space is minimized because very little extra space is in the master records and the number of master records provided for in the data base above the number of records used is kept to a safe minimum. Also, there is an absolute minimum number of linkages in the data base.

Back-up and recovery was made easy by inserting a time-stamp in a master record following its update. This way if the system goes down during an update run, the run need only be resubmitted without regard to the transactions or the data base because the update program always checks to see if it has already updated a record by comparing time-stamps and will not attempt to update a record it has already updated. Expansion of the data base is accomplished by using the back-up and recovery system.

Execution time is minimized for all functions except in the case of extracting, sorting, and writing summaries and/or reports, and this is not critical because of the frequency of these runs.

One can start as simply as we have and gain experience because we have avoided some of the problems of a large data base; such as back-up and recovery efficiency, system complexity, program complexity, disk space management, and maintenance of secondary master files.

One question which was of great concern was: "Will the secondary master files contain all possible keys or only those existing in a variable record?" The implications of this area: (1) If secondary master files contain all possible keys, then an update program would have to be written for each; or, (2) If secondary master files contain only those keys existing in a variable record, then the update program for the primary master file can automatically update the secondary master files as a variable record is written with a key which does not yet exist in the secondary master.

3. THE SYSTEM

3.1 Prompting Inquiries. In this phase, lists of student names and addresses are keypunched. Each type card is read by a module which builds a record which is passed to a general letter writing module. This module prints letters on continuous form letterhead (Appendices 6 and 7) and they are sent out with inquiry reply cards (Appendix 2) which, when returned, get the student on the file as an "inquiry".

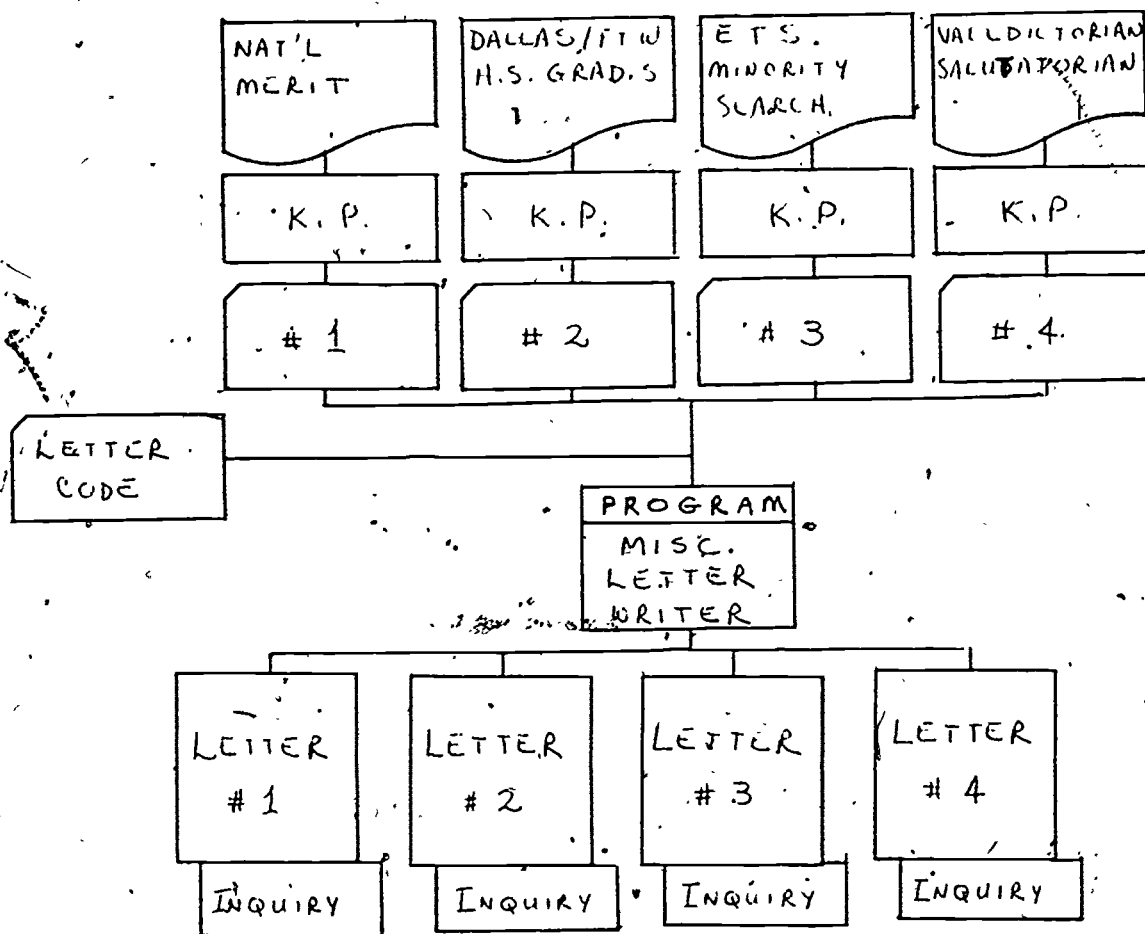


FIGURE 8. System Flow - Prompting Inquiries

3.2 Prompting Concurrent Admission Applications. A program examines the data base and generates follow-up letters (Appendices 6 and 7) for all students who are inquiries. Along with the letter is an application for Concurrent Admission (Appendix 1). Also, mailing labels are written by request to send special literature publications.

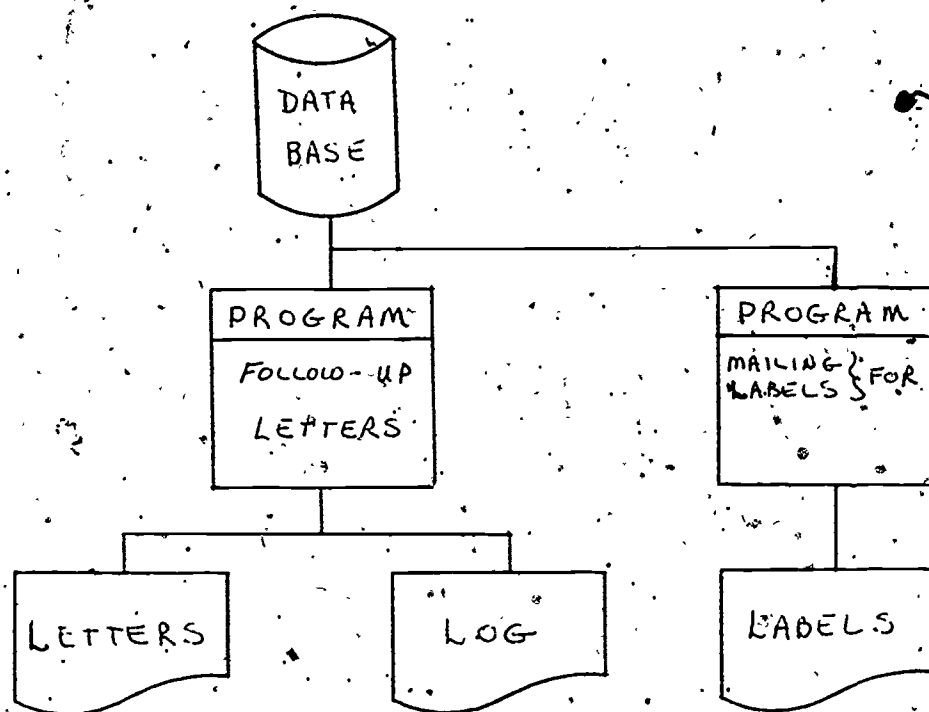


FIGURE 9. System Flow - Prompting Concurrent Admission Applications

3.3 Create and Maintain Concurrent Admission Data Base. This phase is where the data base is built (Figure 10) and all additions, changes, and deletes are accomplished, labels are created for first inquiries, and back-up and recovery procedures are executed.

The data base was created initially by writing a one-time program to dump the old sequential disk file to a temporary disk file in transaction record format (all additions) which was then input to the data base update job. Normally the update job is accomplished by keypunching cards for applications, inquiries, and internally originated file change forms; and processing the weekly accumulation. As inquiry transactions are encountered, a check is done to see if it is the first one for the person, and if so, a mailing label is written. Each time the update job

is run, the back-up and recovery steps pack the transaction records and save them until the next dump of the data base. The data base can be recovered at any time by restoring the last copy, unpacking the saved transactions, and running the update job.

3.4 Using the Data Base. Here applicants are evaluated for admission (Figure 11) by a program which accesses all records in the data base and applies the admission algorithm to all student records which are unevaluated or have been rejected. It uses an unlinked TOTAL master file for obtaining a copy of the acceptance and/or rejection letter that it writes. It writes mailing labels for all students who are accepted, for sending additional pertinent literature. It provides a log of all action taken, and creates a transaction file for the update program which then updates the student's status.

There is a program which will list or summarize the file on any six fields in the master record in any order.

There is a mailing label program which will print a set of labels for a specified group of records.

There are other uses, but those mentioned above are the primary ones.

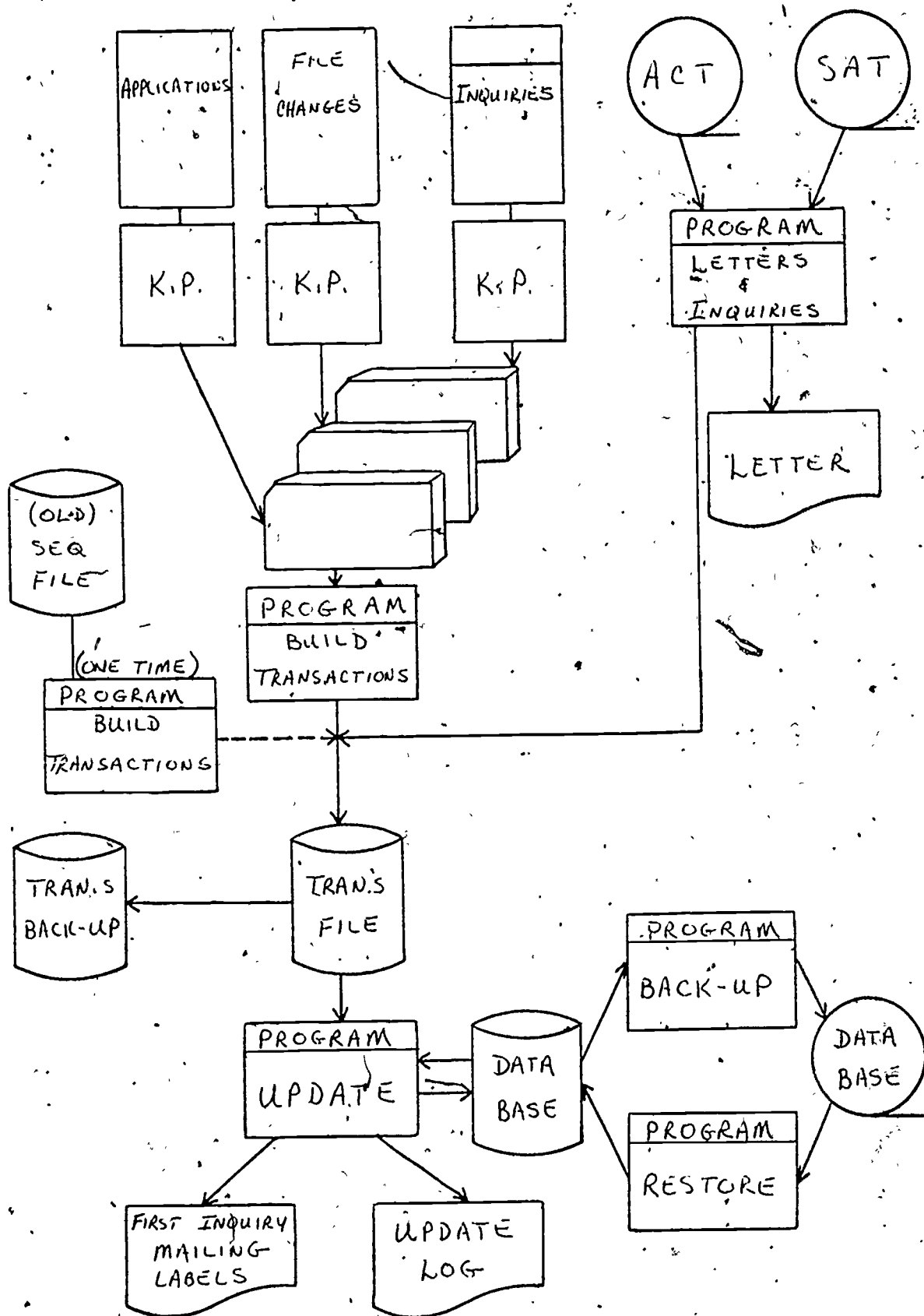


FIGURE 10. System Flow - Create and Maintain Concurrent Admission Data Base

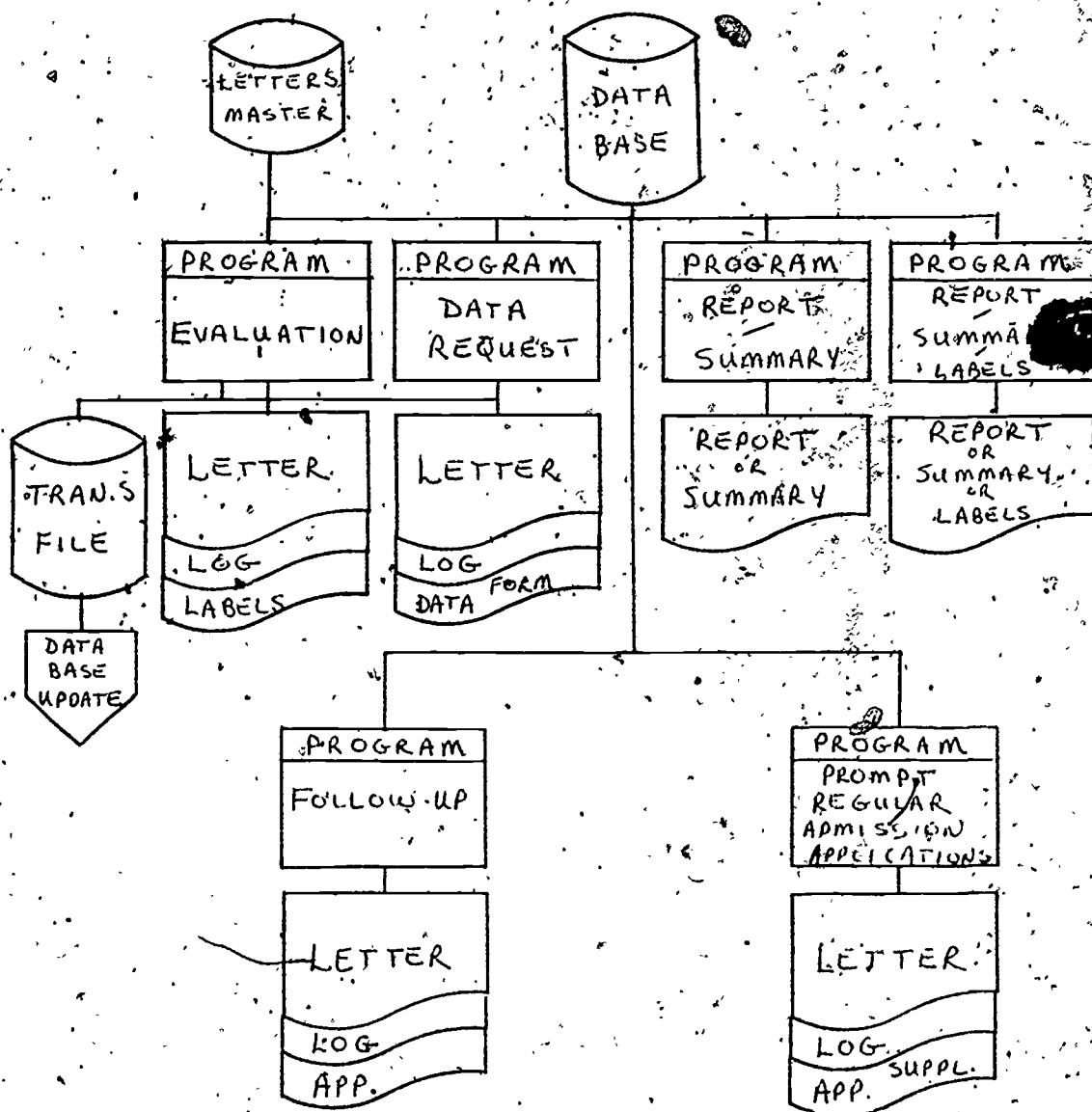


FIGURE 11. System Flow - Using the Data Base

3.5 Prompting Regular Admission Applications. Prompting of regular admission applications by the system is done by a program which accesses all records in the data base and examines the expected date of enrollment to find those scheduled to enroll in the coming semester. It writes a letter to all these with an application and makes a log of all action taken.

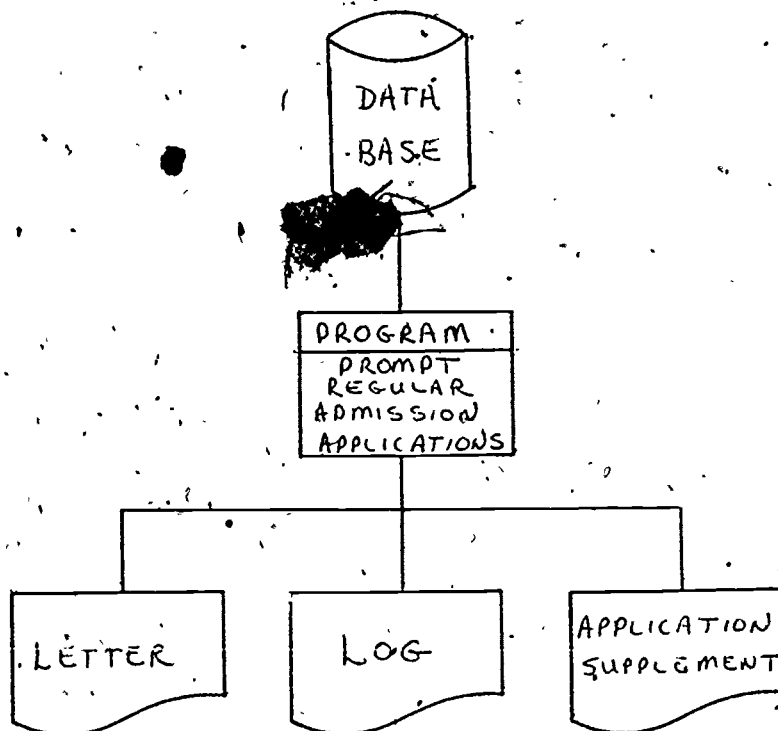


FIGURE 12. System Flow - Prompting Regular Admission Applications

4. IMPLEMENTATION

The implementation plan included maintaining a partially developed sequential file system, while the remaining parts of the system were developed for a Data Base Management System. Then the original programs were converted to run from the data base.

The initial system development using the Data Base Management System consisted of programs to: (1) dump the existing file to transaction format which could be read by the new update program to build the data base initially; (2) update (build) the data base; (3) back-up and recover the data base; and, (4) evaluate applicants for concurrent admission. Then the Detail List/Summary/Label Job was converted to run from Data Base and the conversion was completed.

APPENDIX I.

APPLICATION FOR CONCURRENT ADMISSION
THE UNIVERSITY OF TEXAS AT DALLAS

PART I — TO BE COMPLETED BY ALL APPLICANTS

APPLICANT NAME: LAST FIRST MIDDLE INIT SOCIAL SEC NO

LEGAL RESIDENCE (INCLUDE STREET AND NUMBER CITY)
COUNTY STATE COUNTRY AND ZIP CODE IF USA

CURRENT MAILING ADDRESS (STREET AND NUMBER CITY)
STATE TELEPHONE AND ZIP CODE

EXPECTED SEMESTER OF FIRST ENROLLMENT: ☐ FALL ☐ SPRG ☐ SUM 1 ☐ SUM 2 ☐ YR

MAJOR OR PROGRAM OF STUDY

ATTENDANCE (FULL PART) (DAY NIGHT): ☐ 1-FULL ☐ 2-PART ☐ 1-DAY ☐ 2-NIGHT

COUNTRY OF CITIZENSHIP

DATE OF BIRTH (MO—DAY—YR) SEX: ☐ M ☐ F AGE

HIGH SCHOOL (NAME CITY AND STATE)

INSTITUTION MOST RECENTLY ATTENDED OR ADMITTED (NAME CITY AND STATE)

WHAT ARE YOUR TEST SCORES? (ENTER COMPOSITE SCORES ONLY): ☐ ACT OR ☐ SAT

ARE YOU MARRIED? ☐ YES ☐ NO NUMBER OF DEPENDENT CHILDREN NUMBER UNDER SIX YEARS

ARE YOU A VETERAN? ☐ YES ☐ NO ARE YOU RECEIVING FINANCIAL ASSISTANCE? ☐ YES ☐ NO

ARE YOU PRESENTLY EMPLOYED? ☐ YES ☐ NO WILL YOU NEED ASSISTANCE AT UTD? ☐ 1-YES ☐ 2-NO

IF SO, WHAT TYPE? ☐ 1-LOAN ☐ 2-WORK ☐ 3-SCHOLARSHIP

PLEASE CHECK ONE OF THE FOLLOWING:

☐ 1-AMERICAN NEGRO ☐ 2-AMERICAN INDIAN ☐ 3-AMERICAN SPANISH

☐ 4-AMERICAN ORIENTAL ☐ 5-AMERICAN CAUCASIAN ☐ 6-FOREIGN

UTD USE ONLY

ALPHA NO

LEGAL RES

PROG NO

COUNTRY

HIGH SCH

INST NO

ACT

SAT

ENT CODE

PART II — APPLICANTS WITH QUALIFYING HIGH SCHOOL RANK AND SCORES

DID YOU GRADUATE? ☐ 1-YES ☐ 2-NO WHEN? (MO—DAY—YR)

INDICATE RANK IN CLASS ☐ 1-TOP 10% ☐ 2-TOP 25% ☐ 3-TOP 50%

NOTE TO STUDENT HAVE YOU ENTERED YOUR TEST SCORES IN PART I ABOVE?

HAVE YOU NOTIFIED THE APPROPRIATE TESTING SERVICE TO SEND A REPORT TO UT DALLAS?

PART III — APPLICANTS WITH QUALIFYING COLLEGE CREDITS

INSTITUTION MOST RECENTLY ATTENDED (NAME CITY AND STATE)

NUMBER OF SEMESTER CREDIT HOURS EARNED AT THE ABOVE INSTITUTION ☐ CUMULATIVE GPA AT THE ABOVE INSTITUTION ☐ SYSTEM

HAVE YOU ATTEMPTED WORK ELSEWHERE? ☐ 1-YES ☐ 2-NO IF SO GIVE OVERALL GPA (ALL WORK AT ALL SCHOOLS) ☐

NUMBER OF SEMESTER CREDIT HOURS EARNED AT ALL SCHOOLS ☐

I HEREBY CERTIFY THAT THIS INFORMATION IS COMPLETE AND ACCURATE

APPLICANT'S SIGNATURE

A SIMPLE DATA BASE STRUCTURE FOR
ON-LINE ADMISSIONS AND REGISTRATION

Timothy David R. Martin
Director
Computer Services
Tulsa Junior College

A Simple Data Base Structure for
On-Line Admissions and Registration

Timothy David R. Martin

Director of Computer Services
Tulsa Junior College
Tulsa, Oklahoma

The Data Base Design can determine the success or failure of On-line systems. Data Base structures must be reliable and responsive during the heat of en masse registration.

This paper presents a reliable yet relatively simple Data Base structure designed for on-line Admissions and Registration which is quite adaptable to other applications.

1. INTRODUCTION

More colleges and universities are using COST vs. PRODUCT evaluation techniques than ever before. Costs are being matched with programs and budgets are being cut or raised based on these projected program costs. Everyone feels the pressures of optimizing classroom, building, faculty, and staff resources. Registrars are thus becoming more aware of vacant seats in the various programs and realize that classroom seats are very much like airline seats. If they are not filled at the time of take-off they are lost forever . . . A perishable product. On-line Admissions and Registration procedures are then only a means to reduce the number of seats that perish each semester because of sluggish admissions or untimely reporting.

With all its beauty comes the thorn . . . the possibility of data loss during en masse On-line Registration. The success of the system will, therefore, depend upon the reliability of the Data Base.

This paper discusses a relatively simple Data Base structure which allows On-line Admissions and Registration. The Data Base keys on social security number and permits complete permanent records to be added and maintained while students are being registered in classes.

2. THE MASTER FILE DILEMMA

In many instances qualified applicants walk in with application in hand only hours before the registration process is to close. In an effort to serve the prospective student as well as eliminate those perishable seats it is necessary to perform the following.

2. THE MASTER FILE DILEMMA (Continued)

- (A) Accept the application (if possible)
- (B) Inform the student as to registration procedures (issue time-card, etc.)
- (C) Code application
- (D) Enter application on system
- (E) Register student.

It is at this point that most of the managers of Data Processing who are running On-line shops, (all input into the system coming from On-line applications) have faced a problem which might be called the "Master File Dilemma", briefly stated as follows.

- (A) Is it best to add records On-line to the Master File and risk loss of data

OR

- (B) Capture the data and run in batch mode after the On-line system is down.

Obviously the second technique is the safest, but what about those late applicants and those empty seats in the classroom?

In order to safely add records and quickly retrieve them, we felt standard Indexed Sequential Access Method had to go. It was much too slow in adding records during heavy Admissions and was none too predictable.

The new access method therefore must:

- (A) Add new students On-line and in batch mode simultaneously.
- (B) Have a quick response time.
- (C) Handle multiple records per student.
- (D) Handle multiple files.
- (E) Utilize the present SMF concept.
- (F) Be accessible sequentially.
- (G) Be easy to maintain.

3. THE BASIC RECORD DESIGN

Any institution which attempts to maintain a complete permanent record On-line must allow for new records to be added as transcripts are evaluated and/or classwork is completed. Good Data Base design, therefore, should provide for quick random additions to the file during those very volatile periods of late Admissions and Registration.

3.1 The Social Security (or Student Number) Appendage. One easy method of allowing multiple records per student is simply extending the student number by two characters. These last two characters can be given special meaning such as:

Social Security Number	+	'00'	=	Vital Stat.
Social Security	+	'05'	=	College Transfer
Social Security	+	'59'	=	First Enrollment.

*See Table I for Data Elements.

(NOTICE: Each of the records above '05' have seven logical sub-records.)

3.2 The Key File. The Key File is a relative record file developed to allow for the collisions in randomizing the social security number. The basic social security number is in the form 'AAA-XX-NNNN' where NNNN is a fairly sequential number. The algorithm for finding the key block is as follows:

- (A) Take the last four digits of the S.S.#.
- (B) If greater than 5000 subtract 5000 from the value.
- (C) Take the result from (B) and read that key block.

Each key block is separated into ten sub-records of 44 bytes each.

Each key block will allow ten collisions on same number and eleven records for each student.

DATA BASE MANAGER

RELATIVE
RECORD
"DIRECT"

MASTER
FILES &
OVERFLOW
AREA

RELATIVE
RECORD
"DIRECT"

KEY
FILE

Record
28978

[]

000360319-00 VITAL STATISTICS..

28979

[]

000360319-05 COLLEGE TRANSFER..

28980

[]

000360319-59 CURRENT ENROLLMENT..

28981

[]

000360319-60 CURRENT ENROLLMENT

RECORD NUMBER 0319

KEY 1 | KEY 2 | KEY 3 | . . .

[K-10] 440 BYTES

14 BYTES
60' RELATIVE
RECORD
NUMBER OF
28981

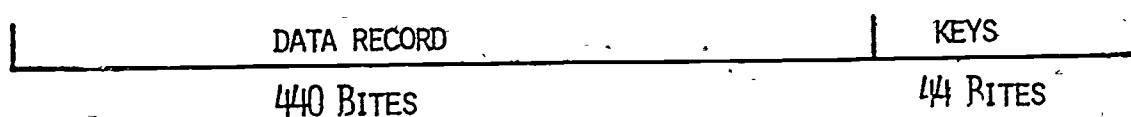
000360319 | . . .

SOCIAL SEC. 60' RELATIVE
RECORD
NUMBER
28978

3.3 The On-Line Master. After the sequential master has been loaded to disk a common overflow area of blank records is provided after the last master record. The relative record location of this first blank record is recorded on the key file under the number '000000000'. Any records that are added are added in the next blank record on the file.

3.4 The Tag-Along Key. To increase response time only one access of the key file is required for each series of reads, rewrites, for a student. After the current key block is located it is appended to the last record read. In doing this the keys become core resident and pointers to every record under each student are available to the Data Management routine.

CORE RESIDENT RECORD



Any additional read or rewrite for this student requires one master file access only.

The new record add command requires the following sequence of I/O's:

- (A) Read key for record 000-00-0000.
- (B) Pick up next overflow pointer.
- (C) Read next overflow record.
- (D) Record should be blank.
- (E) Add record to Master File..
- (F) Update next overflow pointer.
- (G) Re-write 000-00-0000 key record.
- (H) Re-write student record key block.

4. THE COURSE MASTER RECORD

The Course Master File resides within the same physical Disk area as the Basic Student Master File. A common overflow area is shared by all files within the Data Base. All access to the Course Master is then a four byte number called "ZAP". The only difference is the access between the Course Master and Student Master is the algorithm used for finding the key block.

(A) Take right three bytes of Zap Number.

(B) Add 5000.

NOTICE: All key blocks for students range from 1 to 4999. Key blocks for the Course Master range from 5000 to 5999.

5. THE FINAL TEST

The final test of an On-line Admissions and Registration System usually occurs during the heat of registration when every terminal dedicated to registration is backlogged; when the Accounting Office wants their monthly reports; when the Personnel Office wants a payroll run; and when the C.E. wants the system "for just a few minutes". It is during these times that complete confidence in the Data Base is the best Rx available for the Data Processing staff.

Do not be misled in believing that a good solid Data Base will automatically insure a good On-line system. The overall system will only be as innovative as the application programs which access the data. The implementation of

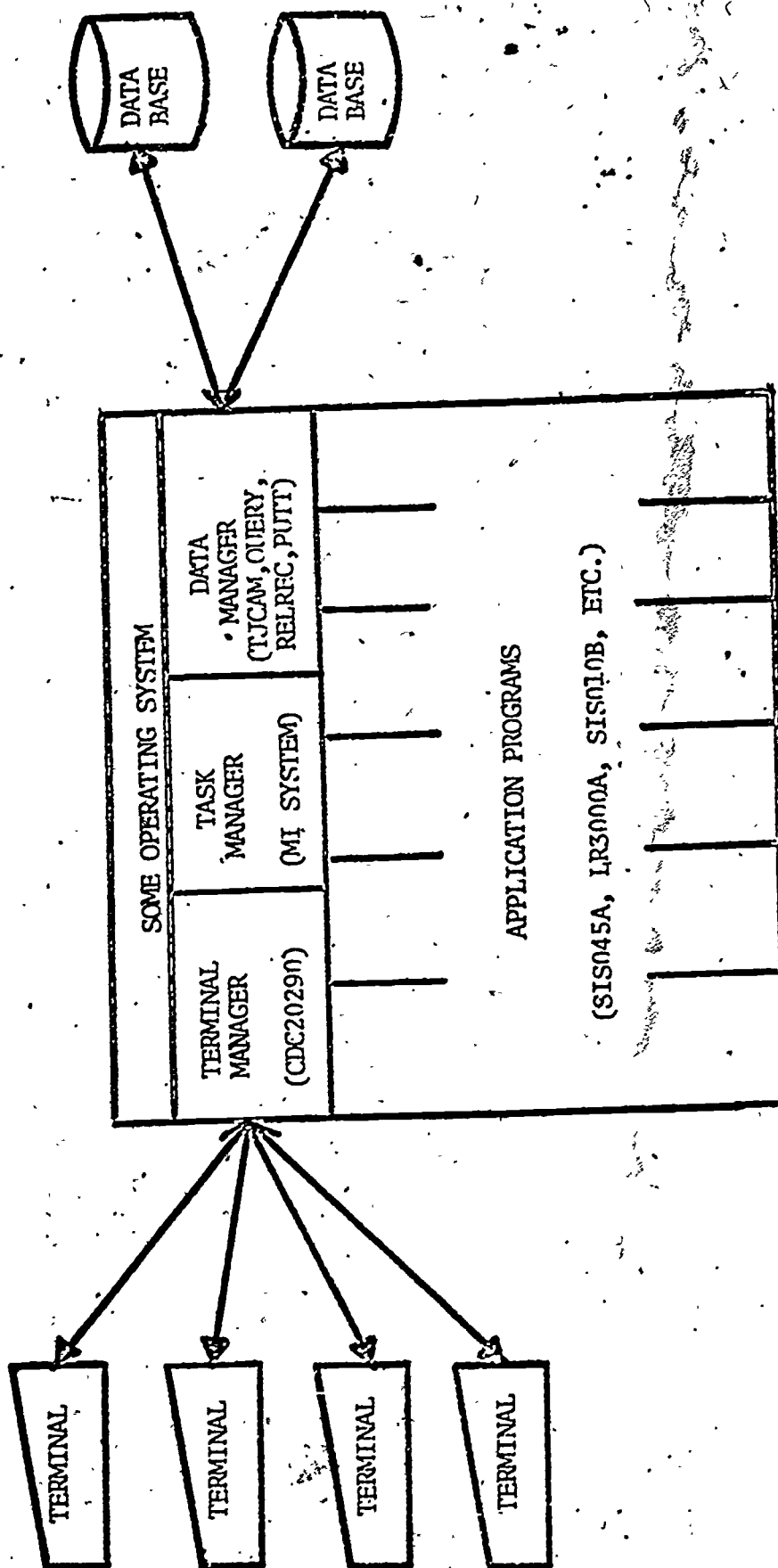
5. THE FINAL TEST (continued)

successful On-line systems rest on the dogged determination to make them succeed. This is quite unlike the man described by Benjamin Franklin:

....., like the man who, in buying an ax of a smith, my neighbor desired to have the whole of it's surface as bright as the edge. The smith consented to grind it bright for him if he would turn the wheel; he turned, while the smith pressed the broad face of the ax hard and heavily on the stone, which made the turning of it very fatiguing. The man came every now and then from the wheel to see how the work went on, and at length would take his ax as it was, without further grinding.

"No", said the smith, "Turn on, turn on; we shall have it bright by and by; as yet, it is only speckled." "Yes", says the man, "but I think I like a speckled ax best."

If the word "speckled" adequately describes the On-line system at your institution the fault may well be in the reliability and responsiveness of the Data Base.



RAPID INQUIRY
MULTIPLE APPLICATIONS
CONCURRENT ACCESS
CONSISTENT DATA

ITEM	FIELD DESCRIPTION	SPACES	POSITION	FORM	OFFICE
1	Paid or delete flag	1	1		
2	Social Security Number	9	2-10	applic.	Admissions
3	Key Field	2	11-12	generated	Computer Ctr.
4	Student Name	19	13-31	applic.	Admissions
5	Street Address	20	32-51	"	"
6	City Name	11	52-62	"	"
7	State (from code table)	2	63-64	"	"
8	Zip Code	5	65-69	"	"
9	Residency (county)	3	70-72	"	"
10	Residency (state)	2	73-74	"	"
11	College Home Phone	7	75-81	"	"
12	Next of Kin Home Phone	10	82-91	"	"
13	Next of Kin	19	92-110	"	"
14	Relationship	1	111	"	"
15	Next of Kin Street Address	20	112-131	"	"
16	Next of Kin City Address	11	132-142	"	"
17	Next of Kin State Code	2	143-144	"	"
18	Next of Kin Zip	5	145-149	"	"
19	Date Of Birth	6	150-155	"	"
20	Place of Birth City	11	156-166	"	"
21	Place of Birth State	2	167-168	"	"
22	Marital Status	1	169	stat.card	Registration
23	Major Interest	3	170-172	appl. advise.	Adm.-Advise
24	Campus Code	1	173	applic.	Admissions
25	Citizenship Code	3	174-176	applic.	"
26	Term of Entrance	3	177-179	applic.	"
27	Attendance	1	180	appl. generated	Adm.&Com. Ctr.
28	F/T-P/T	1	181	"	"
29	Filler	1	182		
30	H.S. Grad, trans, GED, Ind. App.	1	183	applic.	Admissions
31	Date of Graduation	4	184-187	"	"
32	Name of High School	16	188-203	"	"
33	City of High School	11	204-214	"	"
34	County of High School	3	215-217	"	"
35	State of High School	2	218-219	"	"
36	Filler	2	220-221		
37	ACT Scores	10	222-231	Adm&ACT	Adm. & Advise
38	Sex	1	232	Applic.	Admissions
39	Race	1	233	stat.card	Registration
40		2	234-235		
41	Veteran	1	236	Vet.Office	Vet.Claims
42	Filler	11	237-247		
43	Filler - Hold Codes	1	248		
44	Future College	2	249-250	stat.card	Registration
45	Filler-Highest Degree Earned	1	251		
46	Filler-Desired Degree	1	252		
47	Grad. Cand.	1	253	stat.card	Registration
48	Hrs. Employed	1	254-255		
49	Filler	1	256		
50	Residence Code	1	257	applic.	Admissions
51	Date Application Received	6	258-263	"	"
52	\$5 Fee (Date Received)	6	264-269	"	"
53	Date Major Changed	6	270-275	advisement	Advisement
54	ACT (Date Received)	6	276-281	ACT cards	& Adm.

COURSE MASTER FILE

<u>ITEM</u>	<u>FIELD DESCRIPTION</u>	<u>SPACES</u>	<u>POSITION</u>	<u>FORM</u>	<u>OFFICE</u>
1	Flag	1	1	SemSchedule	Academic Dean
2	ZAP	4	2-5	"	"
3	Key	2	6-7	"	"
4	Dept. & Course #	7	8-14	"	"
5	Standard Course Abbreviation	14	15-28	"	"
6	Building & Room	6	29-34	"	"
7	Days of Week	7	35-41	"	"
8	Meeting time of classes	8	42-49	"	"
9	Building & Room 2	6	50-55	"	"
10	Days of Week 2	7	56-62	"	"
11	Meeting times of classes 2	8	63-70	"	"
12	Hours Credit	2	71-72	"	"
13	Hours Lecture	2	73-74	"	"
14	Hours Lab	2	75-76	"	"
15	Fee Code	1	77	"	Bus. Office
16	Amount of Fee	2	78-79	"	"
17	Lab Req.	1	80	SemSchedule	Academic Dean
18	Semester Year	3	81-83	"	"
19	Campus	1	84	"	"
20	NCHEMS code # (last 4 digits)	6	85-90	"	"
21	Fund	1	91	"	"
22	Class Limit	3	92-94	"	"
23	Class Count	3	95-97	"	"
24	Instructor Name - Last First	15	98-112	"	"
25	Instructor Social Security	9	113-121	"	"
26	Blank	1	122	"	"
27	Filler	2	123-124	"	"
28	Division	1	125	"	"
29	CMF Tally	3	126-128	generated	Adm.-Reg.
30	Repeat Code	1	129	SemSchedule	"
31	Number of Weeks	2	130-131	"	"
32	Filler	318	132-440	"	"